

Security of Modes of Operation and other provably secure cryptographic schemes

Ferdinand Sibleyras

Inria, France

23 October 2020



Symmetric Cryptography

- **Cryptography:** Preserve authenticity and privacy of data.
- **Symmetric:** Assume the existence of a shared **secret** and **random** key.
- **Public channel:** Attacker can record and/or manipulate what is being sent.



Building Blocks

A (trusted) primitive

Resistant to known attacks.



Building Blocks

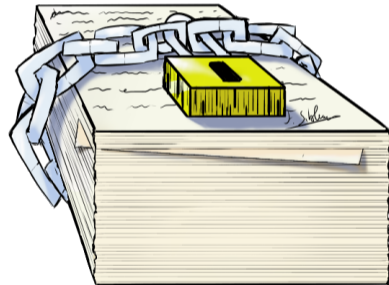
A (trusted) primitive

Resistant to known attacks.

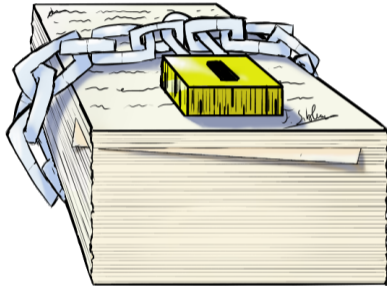


A (proven) mode of operation

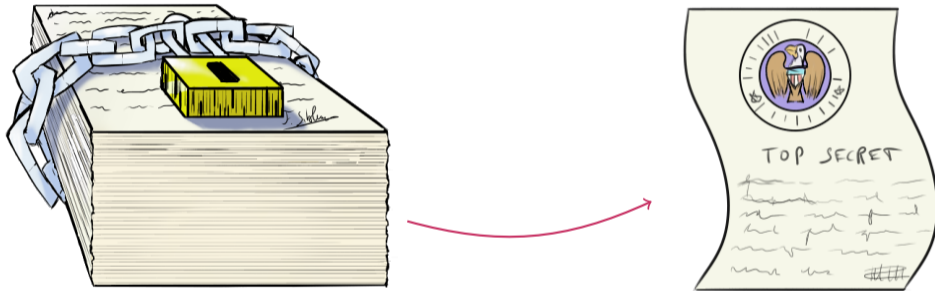
A way to use the primitive.



Security of Modes of Operation



Security of Modes of Operation



Cryptanalysis of Modes

Attacking the mode without attacking the primitive.



Introduction

Block Cipher

$$E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$$

A family of **permutations** indexed by a key where n is the bit size of the permutation or block size.

Example: AES has $n = 128$, 3DES has $n = 64$.



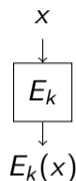
Introduction

Block Cipher

$$E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$$

A family of **permutations** indexed by a key where n is the bit size of the permutation or block size.

Example: AES has $n = 128$, 3DES has $n = 64$.



Introduction

Block Cipher

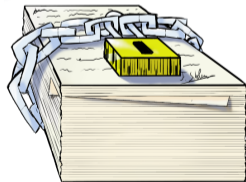
$$E : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$$

A family of **permutations** indexed by a key where n is the bit size of the permutation or block size.

Example: AES has $n = 128$, 3DES has $n = 64$.

Mode of operation

Describes how to use a **block cipher** along with a plaintext message of **arbitrary length** to achieve some concrete cryptographic goals.



Introduction

Modes are classified according to their goals:

- There are **encryption** modes (CBC, CTR, ...). They aim at hiding the plaintext.

Introduction

Modes are classified according to their goals:

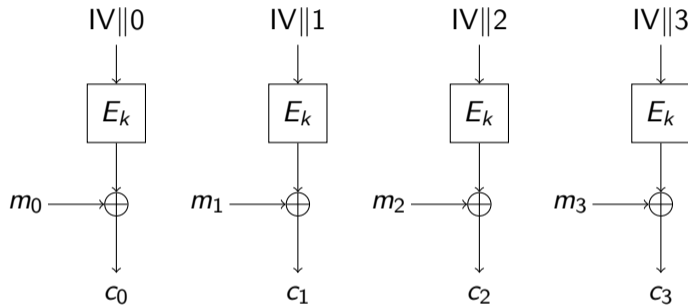
- There are **encryption** modes (CBC, CTR, ...).
They aim at hiding the plaintext.
- There are **authentication** modes (ECBC, GMAC, ...).
They aim at authenticating the plaintext.

Introduction

Modes are classified according to their goals:

- There are **encryption** modes (CBC, CTR, ...).
They aim at hiding the plaintext.
- There are **authentication** modes (ECBC, GMAC, ...).
They aim at authenticating the plaintext.
- There are **authenticated encryption** modes (GCM, SUNDAE, ...).
They aim at both authenticating and hiding the plaintext.

Diagram of the Counter Mode (CTR) [SP 800-38A, 2001]



m_i : The plaintext.

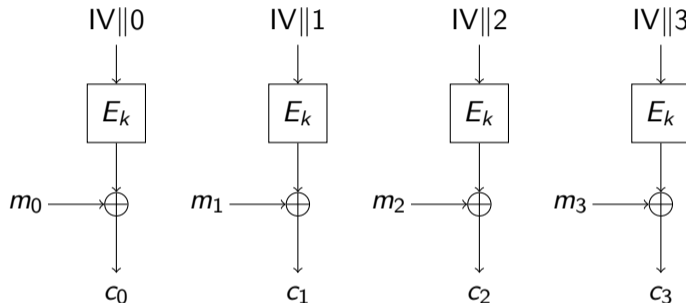
c_i : The ciphertext.

E_k : The block cipher.

IV : The Initialisation Value.

$$c_i = E_k(IV \parallel i) \oplus m_i$$

Diagram of the Counter Mode (CTR) [SP 800-38A, 2001]



m_i : The plaintext.

c_i : The ciphertext.

E_k : The block cipher.

IV : The Initialisation Value.

$$c_i = E_k(IV||i) \oplus m_i$$

★ Akin to a stream cipher: keystream XORed with the plaintext.

★ Inputs $IV||i$ to the block cipher **never repeat**.

The counter mode (CTR) [SP 800-38A, 2001]

Let $K_i = E_k(\text{IV}||i)$ the i th block of keystream.

- If E_k is a good Pseudo-Random Function (PRF) then all K_i are random and this is a one-time-pad.
- A block cipher is a Pseudo-Random **Permutation** (PRP) therefore K_i are all **distinct**: $K_i \neq K_j \forall i \neq j$.

The counter mode (CTR) [SP 800-38A, 2001]

Let $K_i = E_k(\text{IV}||i)$ the i th block of keystream.

- If E_k is a good Pseudo-Random Function (PRF) then all K_i are random and this is a one-time-pad.
- A block cipher is a Pseudo-Random **Permutation** (PRP) therefore K_i are all **distinct**: $K_i \neq K_j \forall i \neq j$.

Distinguishing attack

Truly random keystream blocks would eventually collide.

How many blocks do we expect to gather before we observe a collision ?

Birthday Paradox

Given 30 people, what is the probability that two of them share the same birthday?
more than 70%!

Birthday Paradox

Given 30 people, what is the probability that two of them share the same birthday?
more than 70%!

How many random n -bit words do we expect to collect before a collision ($K_i = K_j$) ?
About $2^{n/2}$ collected blocks!

Security of CTR

Distinguisher of CTR

$\mathcal{O}(2^{n/2})$ blocks of ciphertext are **sufficient** to distinguish CTR from an ideal encryption scheme.

Security of CTR

Distinguisher of CTR

$\mathcal{O}(2^{n/2})$ blocks of ciphertext are **sufficient** to distinguish CTR from an ideal encryption scheme.

Modes of Operation usually comes with a **proof**:

Security proof (σ the number of blocks)

$$\mathbf{Adv}_{\text{CTR}-E}^{\text{IND\$-CPA-N}}(\sigma) \leq \mathbf{Adv}_E^{\text{prf}}(\sigma) \leq \mathbf{Adv}_E^{\text{prp}} + \sigma^2/2^{n+1}$$

Proof of CTR




$\Omega(2^{n/2})$ blocks of ciphertext are **necessary** to distinguish CTR from an ideal encryption scheme.

\implies The security of CTR is **tight** at birthday bound.

Publications – in this talk

-  “The Missing Difference Problem, and Its Applications to Counter Mode Encryption”
Gaëtan Leurent, Ferdinand Sibleyras; **EUROCRYPT 2018**
-  “Low-Memory Attacks Against Two-Round Even-Mansour Using the 3-XOR Problem”
Gaëtan Leurent, Ferdinand Sibleyras; **CRYPTO 2019**
-  “Generic Attacks Against Beyond-Birthday-Bound MACs”
Gaëtan Leurent, Mridul Nandi, Ferdinand Sibleyras; **CRYPTO 2018**
-  “Release of Unverified Plaintext: Tight Unified Model and Application to ANYDAE”
Donghoon Chang, Nilanjan Datta, Avijit Dutta, Bart Mennink, Mridul Nandi, Somitra Sanadhya, Ferdinand Sibleyras; **ToSC 2019**

Publications – other

-  “Generic Attack on Iterated Tweakable FX Constructions”
Ferdinand Sibleyras; **CT-RSA 2020**
-  “On the Security Margin of TinyJAMBU with Refined Differential and Linear Cryptanalysis”
Dhiman Saha, Yu Sasaki, Danping Shi, Ferdinand Sibleyras, Siwei Sun, Yingjie Zhang; **ToSC 2020**
-  “New results on Gimli: full-permutation distinguishers and improved collisions”
Antonio Flórez-Gutiérrez, Gaëtan Leurent, María Naya-Plasencia, Léo Perrin, André Schrottenloher, Ferdinand Sibleyras; **ASIACRYPT 2020**

Organisation

Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

Organisation

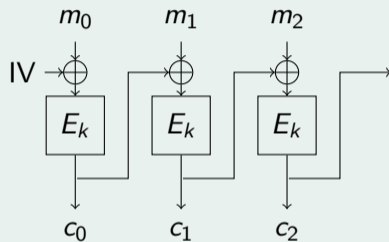
Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

CBC and CTR

Both modes are:

- widely deployed
- proven secure up to birthday bound
- Distinguisher when nearing the bound

CBC mode



Folklore assumptions [Ferguson, Schneier, Kohno]

CTR leaks *very little data*. [...] It would be reasonable to limit the cipher mode to 2^{60} blocks, which allows you to encrypt 2^{64} bytes but restricts the leakage to a small fraction of a bit. When *using CBC mode you should be a bit more restrictive*. [...] We suggest limiting CBC encryption to 2^{32} blocks or so.

Beyond the Distinguisher

From a **distinguishing** attack to a **plaintext recovery** attack?

- If we know m_i , we recover $K_i = c_i \oplus m_i$.
- We can observe repeated encryptions of a secret S that is $c_j = K_j \oplus S$ for many different j .
- The distinguishing attack uses $K_i \oplus K_j \neq 0$ which implies $K_i \oplus c_j \neq S \forall i \neq j$.

Beyond the Distinguisher

From a **distinguishing** attack to a **plaintext recovery** attack?

- If we know m_i , we recover $K_i = c_i \oplus m_i$.
- We can observe repeated encryptions of a secret S that is $c_j = K_j \oplus S$ for many different j .
- The distinguishing attack uses $K_i \oplus K_j \neq 0$ which implies $K_i \oplus c_j \neq S \forall i \neq j$.

Main Idea

Collect many keystream blocks K_i and encryptions of secret block $c_j = K_j \oplus S$; then look for a value s such that $K_i \oplus c_j \neq s \forall i \neq j$.

Missing difference problem

The missing difference problem

- Given \mathcal{A} and \mathcal{B} , and a hint \mathcal{S} three sets of n -bit words
- Find $S \in \mathcal{S}$ such that:

$$\forall (a, b) \in \mathcal{A} \times \mathcal{B}, S \neq a \oplus b.$$

Missing difference problem

Main Idea

Collect many keystream blocks $K_i \in \mathcal{A}$ and encryptions of secret block $c_j = K_j \oplus S \in \mathcal{B}$; then look for a value $s \in \mathcal{S}$ such that $\forall (a, b) \in \mathcal{A} \times \mathcal{B}, s \neq a \oplus b$.

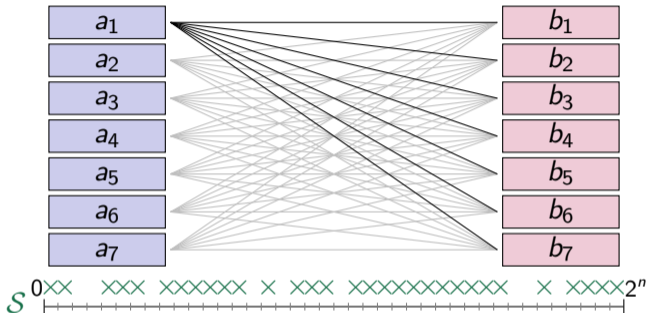
The missing difference problem

- Given \mathcal{A} and \mathcal{B} , and a hint \mathcal{S} three sets of n -bit words
- Find $S \in \mathcal{S}$ such that:

$$\forall (a, b) \in \mathcal{A} \times \mathcal{B}, S \neq a \oplus b.$$

Simple Sieving Algorithm

[McGrew, FSE'13]



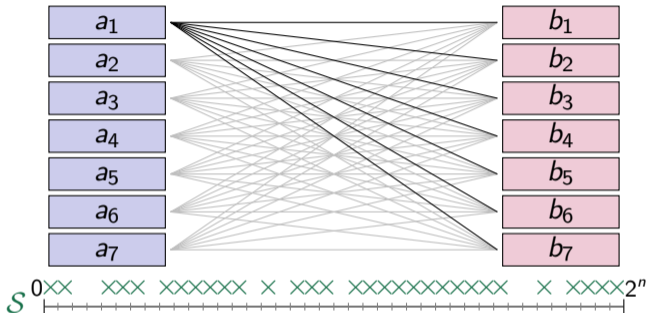
Compute all $a_i \oplus b_j$, remove results from a sieve \mathcal{S} .

Analysis: case $|\mathcal{S}| = 2^n$ via coupon collector problem

- To exclude 2^n candidates of \mathcal{S} , we need $n \cdot 2^n$ values $a_i \oplus b_j$
 - Lists \mathcal{A} and \mathcal{B} of size $\sqrt{n} \cdot 2^{n/2}$. Complexity: $\tilde{O}(2^n)$

Simple Sieving Algorithm

[McGrew, FSE'13]

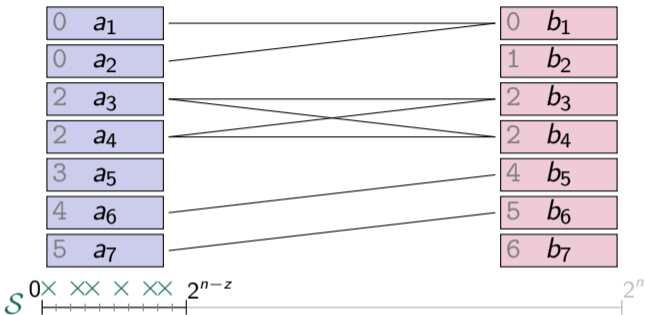


Compute all $a_i \oplus b_j$, remove results from a sieve \mathcal{S} .

Analysis: case $|\mathcal{S}| = 2$

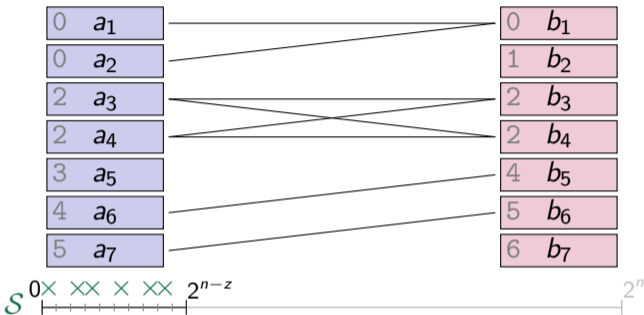
- To exclude **1 candidate** of \mathcal{S} , we need 2^n values $a_i \oplus b_j$
 - Lists \mathcal{A} and \mathcal{B} of size $2^{n/2}$. **Complexity:** $\tilde{O}(2^n)$

Known-prefix Sieving



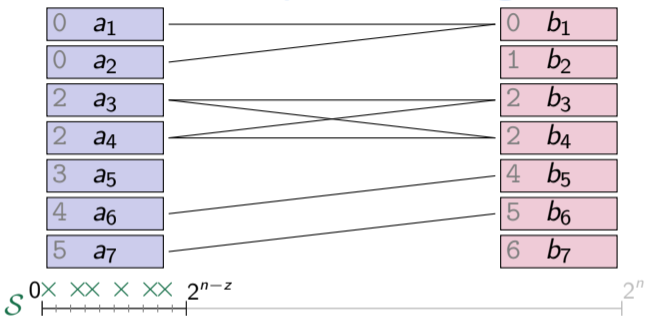
- Assume S starts with z zero bits (more generally, $\dim\langle S \rangle = n - z$)
- Sort lists, consider a_i 's and b_j 's with matching z -bit prefix

Known-prefix Sieving



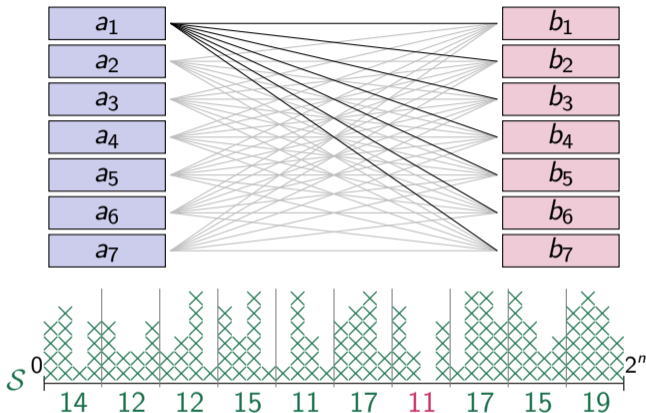
- Assume S starts with z zero bits (more generally, $\dim\langle S \rangle = n - z$)
- Sort lists, consider a_i 's and b_j 's with matching z -bit prefix
- Complexity: $\tilde{O}(2^{n/2} + 2^{\dim\langle S \rangle})$
 - Looking for collision + needed number of collisions

Known-prefix Sieving



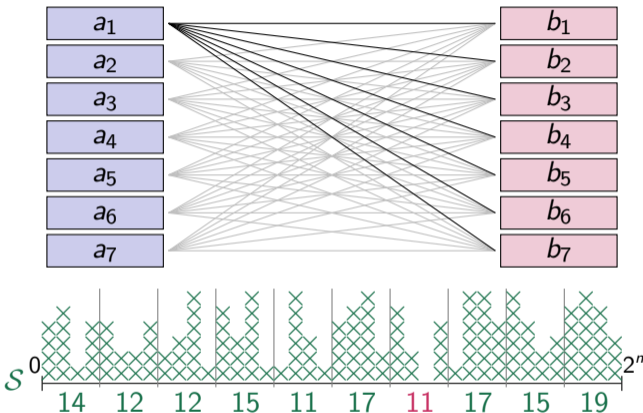
- Assume S starts with z zero bits (more generally, $\dim\langle S \rangle = n - z$)
- Sort lists, consider a_i 's and b_j 's with matching z -bit prefix
- Complexity: $\tilde{O}(2^{n/2} + 2^{\dim\langle S \rangle})$
 - Looking for collision + needed number of collisions
- Complexity: $\tilde{O}(2^{n/2})$ when $\dim\langle S \rangle \leq n/2$

Fast Convolution Sieving



- Instead of computing full sieve, use **buckets** (ie. truncate)

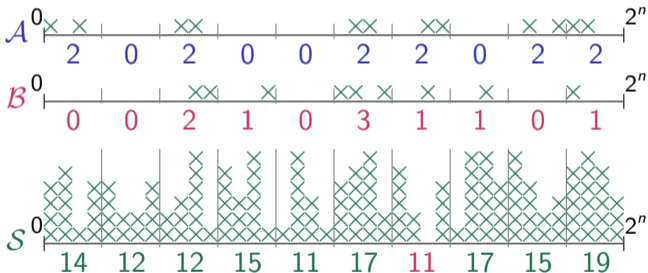
Fast Convolution Sieving



- Instead of computing full sieve, use **buckets** (ie. truncate)
- With enough data, missing difference has **smallest bucket** with high probability
 - Eg. $2^{2n/3}$ queries, sieving with $2^{2n/3}$ buckets of $2^{n/3}$ elements

Computing the sieve

- Count buckets for \mathcal{A} and \mathcal{B}
 - $C_{\mathcal{X}}[i] = |\{x \in \mathcal{X} \mid T(x) = i\}|$



Computing the sieve

- Count buckets for \mathcal{A} and \mathcal{B}
 - $C_{\mathcal{X}}[i] = |\{x \in \mathcal{X} \mid T(x) = i\}|$
 - $C_{\mathcal{S}}[i] = |\{(a, b) \in \mathcal{A} \times \mathcal{B} \mid T(a \oplus b) = i\}|$

$$= \sum_{a \in \mathcal{A}} |\{b \in \mathcal{B} \mid T(a \oplus b) = i\}|$$

$$= \sum_{a \in \mathcal{A}} C_{\mathcal{B}}[i \oplus T(a)]$$

$$= \sum_{j \in \{0,1\}^{n-t}} C_{\mathcal{A}}[j] \cdot C_{\mathcal{B}}[i \oplus j]$$

Computing the sieve

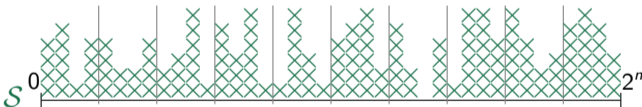
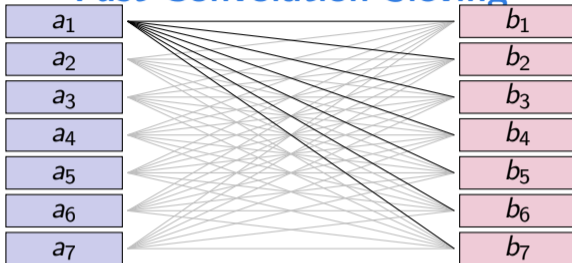
- Count buckets for \mathcal{A} and \mathcal{B}
 - $C_{\mathcal{X}}[i] = |\{x \in \mathcal{X} \mid T(x) = i\}|$
 - $C_{\mathcal{S}}[i] = |\{(a, b) \in \mathcal{A} \times \mathcal{B} \mid T(a \oplus b) = i\}|$

$$= \sum_{a \in \mathcal{A}} |\{b \in \mathcal{B} \mid T(a \oplus b) = i\}|$$

$$= \sum_{a \in \mathcal{A}} C_{\mathcal{B}}[i \oplus T(a)]$$

$$= \sum_{j \in \{0,1\}^{n-t}} C_{\mathcal{A}}[j] \cdot C_{\mathcal{B}}[i \oplus j]$$
- Discrete convolution can be computed efficiently with the Fast Walsh-Hadamard transform!
 - **Complexity:** $\tilde{O}(|C_{\mathcal{S}}|)$ for arbitrary \mathcal{S}

Fast Convolution Sieving



$$T(S) \stackrel{?}{=} \operatorname{argmin} C_S[i]$$

And we can recover the rest of S with the Known-prefix Sieving.

- $2^{2n/3}$ queries, sieving with $2^{2n/3}$ buckets of $2^{n/3}$ elements

Practical Security

BEAST Attack Setting [Duong and Rizzo, 2011]



- Attacker has access to the network (eg. public WiFi)
1. Attacker uses JS to generate traffic
 - Tricks victim to malicious site
 - JS makes *cross-origin* requests
 2. Attacker captures encrypted data
 - Chosen plaintext attack
 - Chosen-Prefix Secret-Suffix model
 $H \rightarrow \mathbf{Enc}(H\|S)$

[Hoang et al., Crypto'15]

Application to CTR (CPSS queries)

- Plaintext recovery using the known-prefix sieving algorithm
- Two kind of queries; half-block and full-block headers:



1. Recover S_1 using the first block of each query:

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(H_1 \| S_1)\} \end{array} \right\} \rightarrow \text{Missing difference: } 0 \| (S_1 \oplus H_2).$$

Application to CTR (CPSS queries)

- Plaintext recovery using the known-prefix sieving algorithm
- Two kind of queries; half-block and full-block headers:



1. Recover S_1 using the first block of each query:

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(H_1 \| S_1)\} \end{array} \right\} \rightarrow \text{Missing difference: } 0 \| (S_1 \oplus H_2).$$

2. When S_1 is known, recover S_2 , with Q_2 queries:

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(S_1 \| S_2)\} \end{array} \right\} \rightarrow \text{Missing difference: } (S_1 \oplus H_1) \| (S_2 \oplus H_2).$$

Application to CTR (CPSS queries)

- Plaintext recovery using the known-prefix sieving algorithm
- Two kind of queries; half-block and full-block headers:



- Recover S_1 using the first block of each query:

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(H_1 \| S_1)\} \end{array} \right\} \rightarrow \text{Missing difference: } 0 \| (S_1 \oplus H_2).$$

- When S_1 is known, recover S_2 , with Q_2 queries:

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(S_1 \| S_2)\} \end{array} \right\} \rightarrow \text{Missing difference: } (S_1 \oplus H_1) \| (S_2 \oplus H_2).$$

- When S_2 is known, recover S_3 :

$$\left. \begin{array}{l} \mathcal{A} = \{\mathcal{E}(H_1 \| H_2)\} \\ \mathcal{B} = \{\mathcal{E}(S_2 \| S_3)\} \end{array} \right\} \rightarrow \text{Missing difference: } (S_2 \oplus H_1) \| (S_3 \oplus H_2).$$

- ...

Results

We defined the **missing difference problem** and **improved** the algorithms to solve it in particular for some cases:

Case	Previous	This work	Improved attacks
\mathcal{S} affine subspace of dim $n/2$	$\tilde{O}(2^{3n/4})$	$\tilde{O}(2^{n/2})$	CTR plaintext recovery
No prior info <i>ie.</i> $ \mathcal{S} = 2^n$	$\tilde{O}(2^n)$	$\tilde{O}(2^{2n/3})$	GMAC, Poly1305 universal forgery

Main take away :

Results

We defined the **missing difference problem** and **improved** the algorithms to solve it in particular for some cases:

Case	Previous	This work	Improved attacks
\mathcal{S} affine subspace of dim $n/2$	$\tilde{O}(2^{3n/4})$	$\tilde{O}(2^{n/2})$	CTR plaintext recovery
No prior info <i>ie.</i> $ \mathcal{S} = 2^n$	$\tilde{O}(2^n)$	$\tilde{O}(2^{2n/3})$	GMAC, Poly1305 universal forgery

Main take away :

- CTR mode **not more secure** than CBC.
- **Frequent rekeying** away from birthday bound will prevent these attacks.

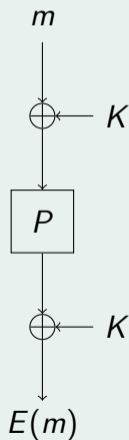
Organisation

Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

1-Round Even-Mansour [Even and Mansour, 1991]

Cryptanalysis in $DQ = DT = 2^n$ originally by Daemen, Asiacrypt 91.

1EM



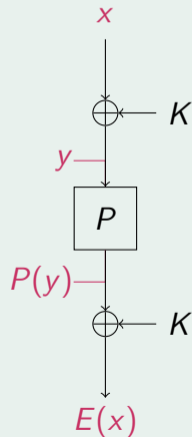
1-Round Even-Mansour [Even and Mansour, 1991]

Cryptanalysis in $DQ = DT = 2^n$ originally by Daemen, Asiacrypt 91.

$\forall x, y \in \{0, 1\}^n,$

$$x \oplus y = K \iff P(y) \oplus E(x) = K$$

1EM



1-Round Even-Mansour [Even and Mansour, 1991]

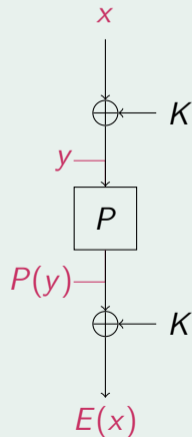
Cryptanalysis in $DQ = DT = 2^n$ originally by Daemen, Asiacrypt 91.

$\forall x, y \in \{0, 1\}^n,$

$$x \oplus y = K \iff P(y) \oplus E(x) = K$$

$$\implies x \oplus E(x) \oplus y \oplus P(y) = 0$$

1EM



1-Round Even-Mansour [Even and Mansour, 1991]

Cryptanalysis in $DQ = DT = 2^n$ originally by Daemen, Asiacrypt 91.

$\forall x, y \in \{0, 1\}^n$,

$$\begin{aligned} x \oplus y = K &\iff P(y) \oplus E(x) = K \\ &\implies x \oplus E(x) \oplus y \oplus P(y) = 0 \end{aligned}$$

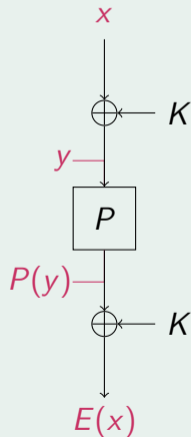
Cryptanalysis via n -bit collision search

Let $f_0(x) = x \oplus E(x)$ and $f_1(y) = y \oplus P(y)$.

Find a collision between f_0 and f_1 , guess $K = x \oplus y$.

\implies **No gap** between the best proofs and attacks.

1EM



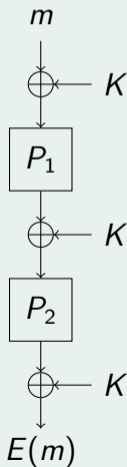
Our Approach

Best information-theoretic attack trade-off: $DQ^2 = 2^{2n}$.

Tight for $D = Q = 2^{2n/3}$.

Best computational attack in $T = 2^n/n$ but it uses also a lot of **memory and/or online data!**

2EM



Our Approach

Best information-theoretic attack trade-off: $DQ^2 = 2^{2n}$.

Tight for $D = Q = 2^{2n/3}$.

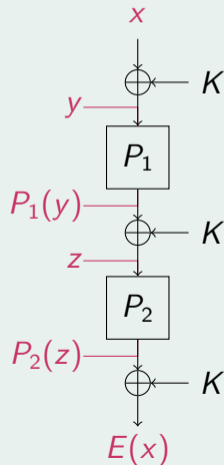
Best computational attack in $T = 2^n/n$ but it uses also a lot of **memory and/or online data!**

In this work, we use the fact that:

$\forall x, y, z \in \{0, 1\}^n$,

$$\begin{cases} x \oplus y = K \\ P_1(y) \oplus z = K \end{cases} \iff \begin{cases} x \oplus y = K \\ P_1(y) \oplus z = K \\ P_2(z) \oplus E(x) = K \end{cases}$$

2EM



Our Approach

Best information-theoretic attack trade-off: $DQ^2 = 2^{2n}$.

Tight for $D = Q = 2^{2n/3}$.

Best computational attack in $T = 2^n/n$ but it uses also a lot of **memory and/or online data!**

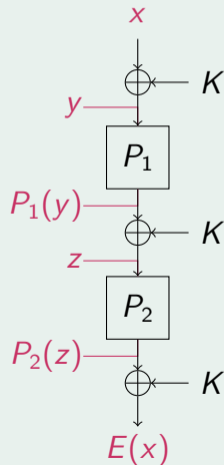
In this work, we use the fact that:

$\forall x, y, z \in \{0, 1\}^n$,

$$\begin{cases} x \oplus y = K \\ P_1(y) \oplus z = K \end{cases} \iff \begin{cases} x \oplus y = K \\ P_1(y) \oplus z = K \\ P_2(z) \oplus E(x) = K \end{cases}$$

$$\implies \begin{cases} x \oplus y \oplus P_1(y) \oplus z = 0 \\ x \oplus E(x) \oplus y \oplus P_2(z) = 0 \end{cases}$$

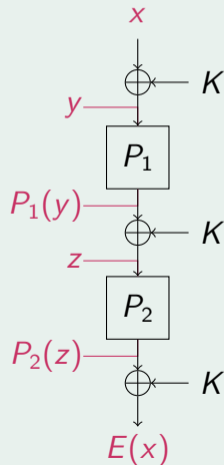
2EM



First result : A Link to the 3-XOR

$$\begin{cases} x \oplus y \oplus P_1(y) \oplus z = 0 \\ x \oplus E(x) \oplus y \oplus P_2(z) = 0 \end{cases}$$

2EM



First result : A Link to the 3-XOR

$$\begin{cases} x \oplus y \oplus P_1(y) \oplus z = 0 \\ x \oplus E(x) \oplus y \oplus P_2(z) = 0 \end{cases}$$

Cryptanalysis via the 3-XOR Problem with $2n$ -bit functions

$$f_0(x) = x \quad || \quad x \oplus E(x)$$

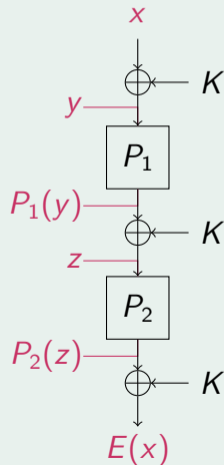
$$f_1(y) = y \oplus P_1(y) \quad || \quad y$$

$$f_2(z) = z \quad || \quad P_2(z)$$

Find x, y, z such that $f_0(x) \oplus f_1(y) \oplus f_2(z) = 0$;

Guess $K = x \oplus y$.

2EM



3-XOR Problem

Definition (Collision problem)

Given two functions f_0, f_1 , find two inputs (x_0, x_1) such that $f_0(x_0) \oplus f_1(x_1) = 0$.

Definition (3-XOR problem)

Given three functions f_0, f_1, f_2 , find three inputs (x_0, x_1, x_2) such that $f_0(x_0) \oplus f_1(x_1) \oplus f_2(x_2) = 0$.

Definition (3-XOR problem with lists)

Given three lists L_0, L_1, L_2 , find three elements $(e_0, e_1, e_2) \in L_0 \times L_1 \times L_2$ such that $e_0 \oplus e_1 \oplus e_2 = 0$.

Strategy

3-XOR solving

Two main techniques:

Multicollision-based [Nikolic&Sasaki15] and Linear algebra-based [Joux09].

Roughly the same asymptotic time complexity.

Strategy

3-XOR solving

Two main techniques:

Multicollision-based [Nikolic&Sasaki15] and Linear algebra-based [Joux09].

Roughly the same asymptotic time complexity.

2EM cryptanalysis

Except for one, [DDKS16], all previous cryptanalysis use multicollision-based techniques.

Exhibiting the link to 3-XOR allows us to deeply explore linear algebra-based techniques for cryptanalysis.

Benefits : Reduced online complexity AND memory both arguably costlier than time.

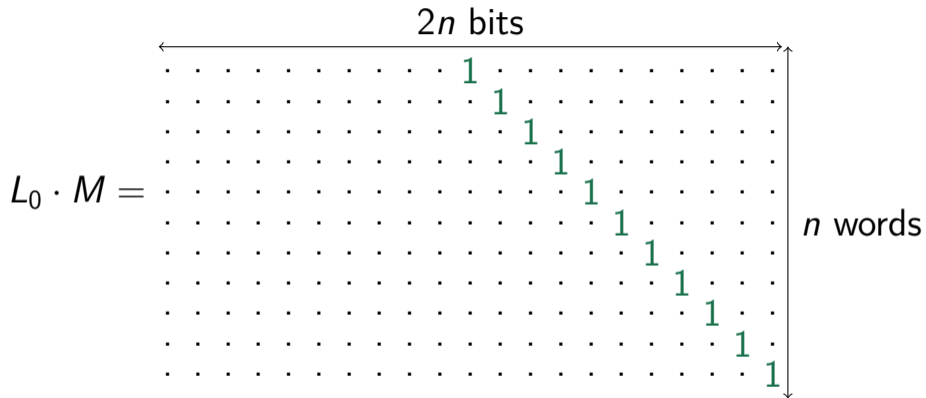
Joux's Technique [Joux, 2009]

$2n$ bits

$$L_0 = \begin{array}{cccccccccccccccccccc} 1 & \cdot & 1 & \cdot & 1 & 1 & \cdot & 1 & 1 & \cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 \\ \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \cdot & 1 & \cdot & 1 \\ 1 & 1 & \cdot & \cdot & 1 & 1 & 1 & \cdot & 1 & 1 & \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 \\ 1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & 1 & 1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot & 1 & 1 & 1 \\ 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & 1 & 1 & 1 & \cdot & \cdot & 1 \\ 1 & \cdot & \cdot & 1 & 1 & \cdot & \cdot & 1 & 1 & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 & 1 & \cdot & \cdot & \cdot & 1 \\ \cdot & 1 & \cdot & 1 & 1 & \cdot & \cdot & 1 & 1 & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 & 1 & \cdot & \cdot & 1 & 1 \\ 1 & \cdot & 1 & 1 & 1 & 1 & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \cdot & \cdot & \cdot & 1 & \cdot \\ \cdot & \cdot & 1 & 1 & 1 & 1 & 1 & 1 & \cdot & 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot \end{array}$$

n words

Joux's Technique [Joux, 2009]



$$e_0 \oplus e_1 \oplus e_2 = 0 \iff e_0 \cdot M \oplus e_1 \cdot M \oplus e_2 \cdot M = 0$$

$$3\text{-XOR with } L_0, L_1, L_2 \iff 3\text{-XOR with } L_0 \cdot M, L_1 \cdot M, L_2 \cdot M$$

Joux's Technique [Joux, 2009]

1. Compute M s.t. $L_0 \cdot M = 0_n || I_n$;
2. $L'_1 = L_1 \cdot M$;
3. $L'_2 = L_2 \cdot M$;
4. Look for partial n -bit collisions between L'_1 and L'_2 ;
5. Check if Solution.

Complexity

$$|L_0| = n$$

$$|L_1| = |L_2| = \frac{2^n}{\sqrt{n}}$$

$$\implies |L_0| \cdot |L_1| \cdot |L_2| = 2^{2n} \checkmark$$

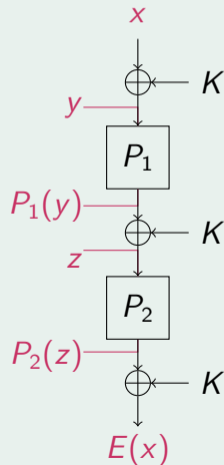
$\mathcal{O}\left(\frac{2^n}{\sqrt{n}}\right)$ memory and computations.

Easy Clamping

We are **NOT** in the **random** 3-XOR case.

- $L_0 \ni x \parallel x \oplus E(x)$
- $L_1 \ni y \oplus P_1(y) \parallel y$
- $L_2 \ni z \parallel P_2(z)$

2EM

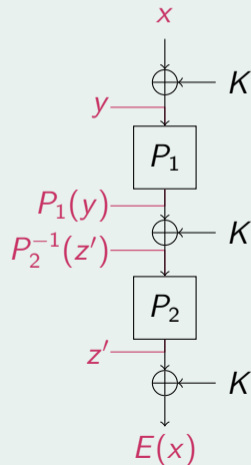


Easy Clamping

We are **NOT** in the **random** 3-XOR case.

1. $L_0 \ni x \parallel x \oplus E(x)$
2. $L_1 \ni y \oplus P_1(y) \parallel y$
3. $L_2 \ni P_2^{-1}(z') \parallel z'$

2EM



Easy Clamping

We are **NOT** in the **random** 3-XOR case.

$$1. L_0 \ni \quad x \quad \parallel \quad x \oplus E(x)$$

$$2. L_1 \ni \quad y \oplus P_1(y) \quad \parallel \quad y$$

$$3. L_2 \ni \quad P_2^{-1}(z') \quad \parallel \quad z'$$

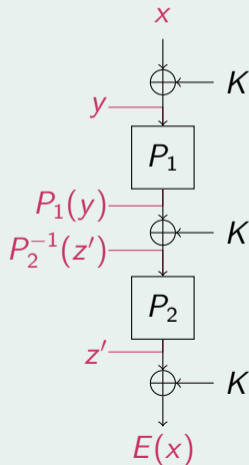
Let $D = 2^d$ thus $Q = 2^{n-d/2}$

$\implies DQ^2 = 2^{2n} \checkmark$

Only compute for y and z' with $d/2$ trailing zeroes.

Only keep $x \oplus E(x)$ with $d/2$ trailing zeroes.

2EM



Easy Clamping

We are **NOT** in the **random** 3-XOR case.

$$1. L_0 \ni \quad x \quad \parallel \quad x \oplus E(x)$$

$$2. L_1 \ni \quad y \oplus P_1(y) \quad \parallel \quad ** | 0..0$$

$$3. L_2 \ni \quad P_2^{-1}(z') \quad \parallel \quad ** | 0..0$$

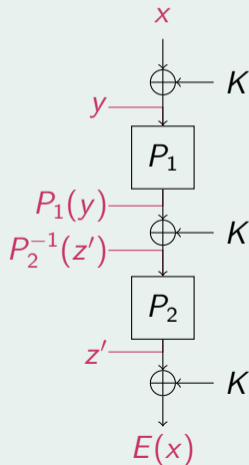
Let $D = 2^d$ thus $Q = 2^{n-d/2}$

$$\implies DQ^2 = 2^{2n} \checkmark$$

Only compute for y and z' with $d/2$ trailing zeroes.

Only keep $x \oplus E(x)$ with $d/2$ trailing zeroes.

2EM



Easy Clamping

We are **NOT** in the **random** 3-XOR case.

- $L_0 \ni x \parallel x \oplus E(x)$
- $L_1 \ni y \oplus P_1(y) \parallel ** | 0..0$
- $L_2 \ni P_2^{-1}(z') \parallel ** | 0..0$

Let $D = 2^d$ thus $Q = 2^{n-d/2} \implies DQ^2 = 2^{2n} \checkmark$

Only compute for y and z' with $d/2$ trailing zeroes.

Only keep $x \oplus E(x)$ with $d/2$ trailing zeroes.

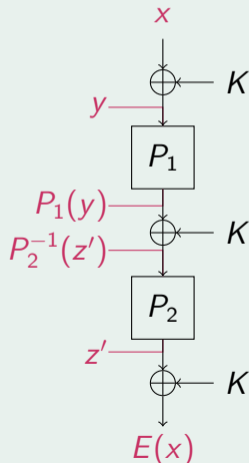
3-XOR after clamping

$$|L_0| = D/2^{d/2} = 2^{d/2}$$

$$|L_1| = |L_2| = Q = 2^{n-d/2}$$

Reduced lists of $2n - d/2$ -bit elements.

2EM



Known 3-XOR Solvers

By [Bouillaguet, Delaplace, Fouque. ToSC 2018] with $w = 2n - d/2$:

Repeat $\mathcal{O}(|L_0|/w)$ times Joux's algorithm.

Realistic 3-XOR algorithm.

$$T = \mathcal{O}(2^n/n)$$

$$M = \mathcal{O}(2^{n-d/2})$$

Revisited Baran-Demaine-Pătraşcu 3-SUM algorithm

Best known asymptotic complexity but impractical for realistic w .

$$T = \mathcal{O}(2^n \cdot \ln^2(n)/n^2)$$

$$M = \mathcal{O}(2^{n-d/2})$$

Strategy	Data	Queries	Time	Memory	Param.
Clamping + BDF algo	2^d KP	$2^{n-d/2}$	$2^n/n$	$2^{n-d/2}$	$0 < d < n$
Clamping + BDP algo	2^d KP	$2^{n-d/2}$	$2^n \ln^2 n/n^2$	$2^{n-d/2}$	$0 < d < n$

Joux's Technique... but smaller

$$L_0^1 || L_0^2 =$$

← 2n bits →																				λn words
·	1	·	·	1	1	·	·	1	·	·	·	·	·	·	·	·	1	·		
1	·	·	·	·	·	1	·	1	1	·	1	1	1	·	·	1	·	·	1	
·	·	·	·	1	1	1	·	1	1	1	·	1	1	1	·	1	1	·	·	
1	1	·	·	1	·	·	1	·	·	1	·	1	1	1	1	1	1	1	1	
1	·	1	1	1	·	·	·	1	·	1	·	·	1	1	1	1	1	1	1	
1	·	·	1	1	1	·	·	1	1	1	1	·	1	·	1	·	·	·	1	

Joux's Technique... but smaller

$$L_0^1 || (L_0^2 \cdot M_s) =$$

←----- 2n bits -----→																				λn words
·	1	·	·	1	1	·	·	1	·	·	·	·	·	·	1	·	·	·	·	
1	·	·	·	·	·	1	·	1	1	·	·	·	·	·	·	1	·	·	·	
·	·	·	·	1	1	1	·	1	1	·	·	·	·	·	·	·	1	·	·	
1	1	·	·	1	·	·	1	·	·	·	·	·	·	·	·	·	·	1	·	
1	·	1	1	1	·	·	1	·	·	·	·	·	·	·	·	·	·	1	·	
1	·	·	1	1	1	·	·	1	1	·	·	·	·	·	·	·	·	·	1	
←----- n -----→				←----- (1 - λ)n -----→							←----- λn -----→									

Joux's Technique... but smaller

$$L_0^1 || (L_0^2 \cdot M_s) =$$

←----- 2n bits ----->																				λn words	
·	1	·	·	1	1	·	·	1	·	·	·	·	·	·	·	·	·	·	·		
1	·	·	·	·	·	1	·	1	1	·	·	·	·	·	·	·	·	·	1		·
·	·	·	·	1	1	1	·	1	1	·	·	·	·	·	·	·	·	·	1		·
1	1	·	·	1	·	·	1	·	·	·	·	·	·	·	·	·	·	·	·		1
1	·	1	1	1	·	·	·	1	·	·	·	·	·	·	·	·	·	·	·		1
1	·	·	1	1	1	·	·	1	1	·	·	·	·	·	·	·	·	·	·	1	
←----- n ----->				←----- (1 - λ)n ----->							←----- λn ----->										

$$L_1 \ni \quad y \oplus P_1(y) \quad || \quad y$$

$$L_2 \ni \quad P_2^{-1}(z') \quad || \quad z'$$

Joux's Technique... but smaller

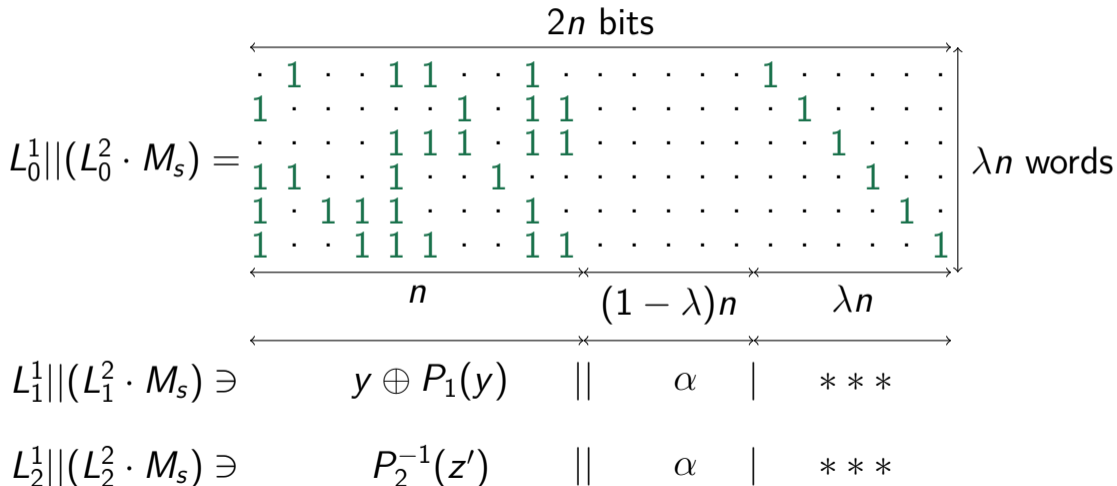
$$L_0^1 || (L_0^2 \cdot M_s) = \begin{matrix} \xleftarrow{2n \text{ bits}} & & \xrightarrow{\lambda n \text{ words}} \\ \cdot & 1 & \cdot & \cdot & 1 & 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\ 1 & 1 & \cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ 1 & \cdot & 1 & 1 & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \\ 1 & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 \end{matrix}$$

$n \qquad (1 - \lambda)n \qquad \lambda n$

$$L_1^1 || (L_1^2 \cdot M_s) \ni \quad y \oplus P_1(y) \quad || \quad y \cdot M_s$$

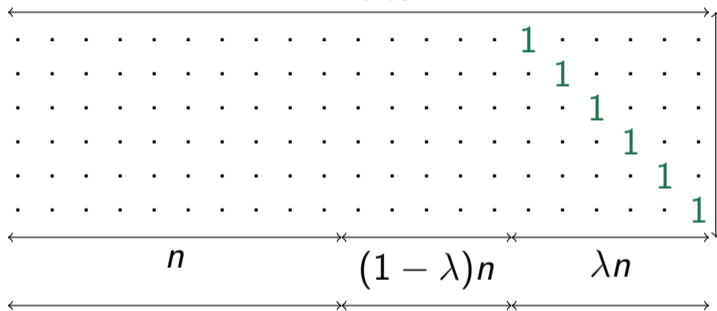
$$L_2^1 || (L_2^2 \cdot M_s) \ni \quad P_2^{-1}(z') \quad || \quad z' \cdot M_s$$

Joux's Technique... but smaller



Joux's Technique... but smaller

2n bits



λn words

n

$(1 - \lambda)n$

λn

$$L_1^1 || (L_1^2 \cdot M_s) \ni$$

$$y \oplus P_1(y)$$

||

α

|

$$L_2^1 || (L_2^2 \cdot M_s) \ni$$

$$P_2^{-1}(z')$$

||

α

|

Low-Data Attack on 2EM

1. Collect λn plaintext/ciphertext pairs for L_0 and compute M_s .
2. Pick a new $(n - \lambda n)$ -bit value α :
 - 2.1 For all λn -bit value u : let $y = z' = (\alpha || u) \cdot M_s^{-1}$ and fill L_1 and L_2 .
 - 2.2 Solve the 3-XOR over L_0, L_1, L_2 using Joux's technique.
(Only an $(n + \lambda n)$ -bit collision)
 - 2.3 Clear L_1 and L_2 . Loop if no solution.
3. Guess $K = x \oplus y$ for the solution found.

L_1 and L_2 contain $2^{\lambda n}$ elements and reused for different α .

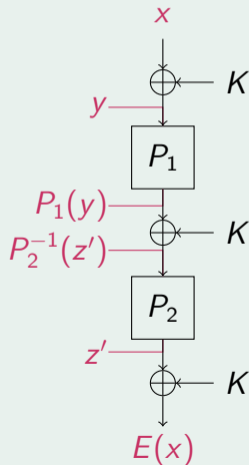
Complexity

Data $D = \lambda n$.

Memory $\mathcal{O}(2^{\lambda n})$.

Time $T = Q = \mathcal{O}(\frac{2^n}{\lambda n})$.

2EM



Results

Ref	Data	Queries	Time	Memory	Param.
[NWW13]	$2^n \ln n/n$ KP	$2^n \ln n/n$	$2^n \ln n/n$	$2^n \ln n/n$	
[DDKS13]	$2^{\lambda n}$ KP	$2^n \ln n/n$	$2^n \ln n/n$	$2^n \ln n/n$	
[DDKS16]	$2^n/\lambda n$ CP	$2^n/\lambda n$	$2^n/\lambda n$	$2^{\lambda n}$	$0 < \lambda < \frac{1}{3}$
[IsoShi17]	$2^n \ln n/n$ CP	$2^n \ln n/n$	$2^n \ln n/n$	$2^n \ln n/n$	
	$2^{\lambda n}$ CP	$2^n \ln n/n$	$2^n \ln n/n$	$2^n \ln n/n$	
	$2^n \beta/n$ CP	$2^n/2^\beta$	$2^n \beta/n$	$2^n/2^\beta$	$\log n \leq \beta < n$
Clamping + BDF	2^d KP	$2^{n-d/2}$	$2^n/n$	$2^{n-d/2}$	$0 < d < n$
Clamping + BDP	2^d KP	$2^{n-d/2}$	$2^n \ln^2 n/n^2$	$2^{n-d/2}$	$0 < d < n$
Low-Data	λn KP	$2^n/\lambda n$	$2^n/\lambda n$	$2^{\lambda n}$	$0 < \lambda < 1$

Organisation

Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

Double-bloc Hash-then-Sum MACs

How to build a deterministic MAC secure **beyond birthday bound**?

DbHtS Strategy

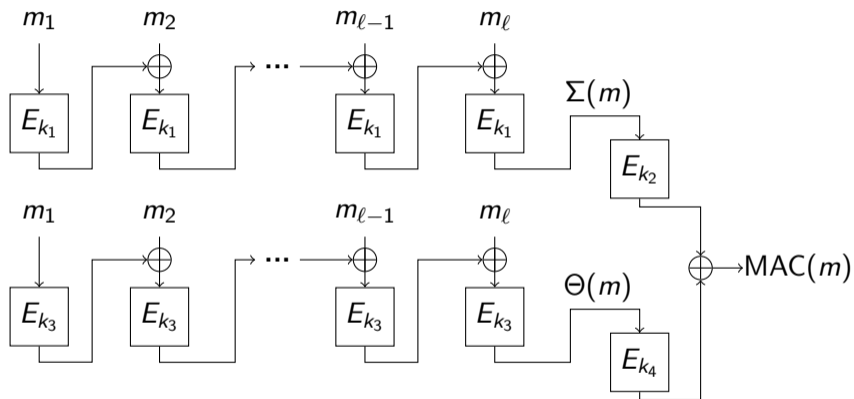
Doubling the state size to $2n$ so that an internal state collision only happens after $\mathcal{O}(2^n)$ processed blocks.

$$\text{MAC}(m) = E(\Sigma(m)) \oplus E'(\Theta(m))$$

Many constructions: SUM-ECBC, GCM-SIV2, PMAC+, LightMAC+, 3kf9, ...

Most DbHtS constructions came with a proof up to $\Omega(2^{2n/3})$ blocks but **no attacks**.

SUM-ECBC [Yasuda, 2010]



$$\text{MAC}(m) = E_{k_2}(\Sigma(m)) \oplus E_{k_4}(\Theta(m))$$

4-way collision for double-hash-then-sum schemes

Look for a quadruple of messages X, Y, Z, T that satisfies:

$$\mathcal{R}(X, Y, Z, T) := \begin{cases} \Sigma(X) = \Sigma(Y) \\ \Theta(Y) = \Theta(Z) \\ \Sigma(Z) = \Sigma(T) \\ \Theta(T) = \Theta(X) \end{cases}$$

$$\mathcal{R}(X, Y, Z, T) \implies \text{MAC}(X) \oplus \text{MAC}(Y) \oplus \text{MAC}(Z) \oplus \text{MAC}(T) = 0$$

$$\begin{array}{ccc} \text{MAC}(X) = E(\Sigma(X)) \oplus E'(\Theta(X)) & \overbrace{E'(\Theta(T)) \oplus E(\Sigma(T))} = & \text{MAC}(T) \\ \parallel & & \parallel \\ \text{MAC}(Y) = E(\Sigma(Y)) \oplus E'(\Theta(Y)) & \underbrace{E'(\Theta(Z)) \oplus E(\Sigma(Z))} = & \text{MAC}(Z) \end{array}$$

4-way collision for double-hash-then-sum schemes

With carefully crafted sets of messages for X, Y, Z, T :

$$\begin{cases} \Sigma(X) = \Sigma(Y) \\ \Theta(Y) = \Theta(Z) \\ \Sigma(Z) = \Sigma(T) \end{cases} \implies \Theta(T) = \Theta(X).$$

Thus $\mathcal{R}(X, Y, Z, T) \iff \begin{cases} \Sigma(X) = \Sigma(Y) \\ \Theta(Y) = \Theta(Z) \\ \Sigma(Z) = \Sigma(T) \end{cases}$ a *3n-bit condition*.

4-way collision for double-hash-then-sum schemes

With carefully crafted sets of messages for X, Y, Z, T :

$$\begin{cases} \Sigma(X) = \Sigma(Y) \\ \Theta(Y) = \Theta(Z) \\ \Sigma(Z) = \Sigma(T) \end{cases} \implies \Theta(T) = \Theta(X).$$

Thus $\mathcal{R}(X, Y, Z, T) \iff \begin{cases} \Sigma(X) = \Sigma(Y) \\ \Theta(Y) = \Theta(Z) \\ \Sigma(Z) = \Sigma(T) \end{cases}$ a $3n$ -bit condition.

Query complexity

There are $\simeq q_t^4$ quadruples for a $3n$ -bit condition.

A good one with high probability after $q_t \simeq 2^{3n/4}$ queries.

Finding a good quadruple

1. Define and build L_0, L_1, L_2, L_3 of size $2^{3n/4}$.
2. Solve the 4-XOR problem on those lists.

Algorithm cost

Step 1 costs $q_t = \mathcal{O}(2^{3n/4})$ queries and as much memory.

Step 2 is about solving an instance of the 4-XOR problem.

Using the memory efficient technique it costs in $\mathcal{O}(2^{3n/4})$ memory and $\mathcal{O}(2^{3n/2})$ time.

Exploiting particularities

When the solutions are related, we can solve the 4-XOR problem in $\mathcal{O}(2^{6n/7})$ queries, time and memory.

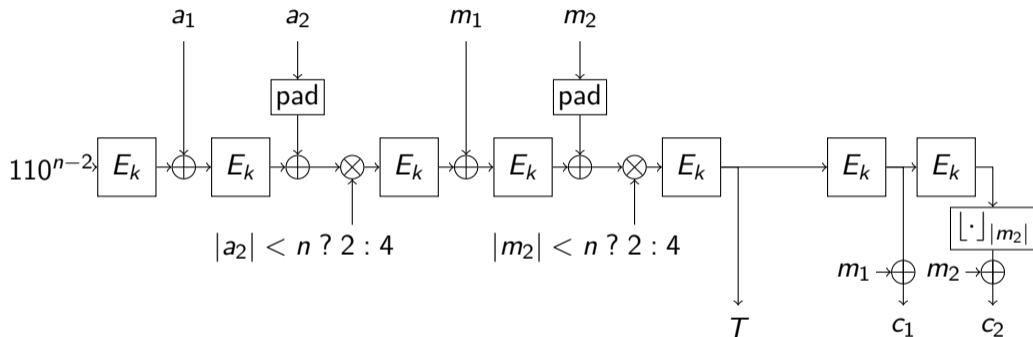
Mode	Proofs Queries	Attacks (this work)		
		Queries	Time	Type
SUM-ECBC	$\Omega(2^{3n/4})$	$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Universal
GCM-SIV2	$\Omega(2^{2n/3})$	$\mathcal{O}(2^{6n/7})$	$\tilde{\mathcal{O}}(2^{6n/7})$	Universal
		$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Universal
PMAC+	$\Omega(2^{3n/4})$	$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Existential
		$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Existential
LightMAC+	$\Omega(2^n)$	$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Existential
	$\Omega(2^{2n/3})$			
1kPMAC+	$\Omega(2^{2n/3})$	$\mathcal{O}(2^{3n/4})$	$\tilde{\mathcal{O}}(2^{3n/2})$	Existential
3kf9	$\Omega(2^{3n/4})$	$\mathcal{O}(\sqrt[4]{n} \cdot 2^{3n/4})$	$\tilde{\mathcal{O}}(2^{5n/4})$	Universal
1kf9	$\Omega(2^{2n/3})$	$\mathcal{O}(2^{n/2})$	$\tilde{\mathcal{O}}(2^{n/2})$	Universal

Later **bounds** [KLL20] proved the optimality of our attack for many schemes. Our attack contradicts a flawed proof of LightMAC+ [RSA:Naito18]. The proof of 1kf9 was known wrong and our attack shows it cannot be fixed.

Organisation

Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

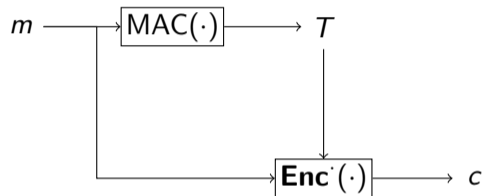
SUNDAE [Banik et al. 2018]



Authenticated Encryption

Encrypt (a, m) into (a, c, T) where T authenticates a and m .

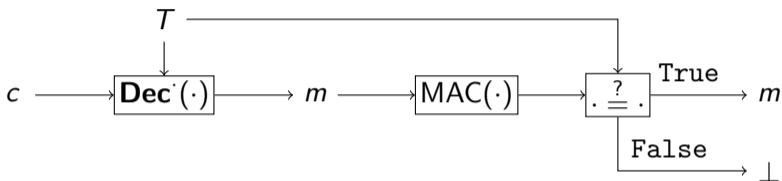
SIV structure



Encryption

Use the output tag as the IV for encryption.

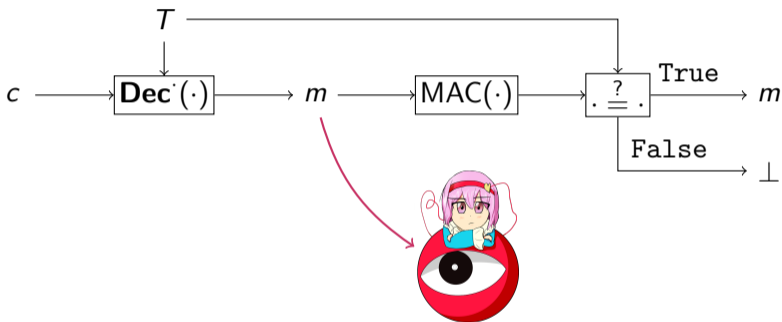
SIV structure



Decryption

Use the received tag as an IV and authenticate again to verify if the tags match.

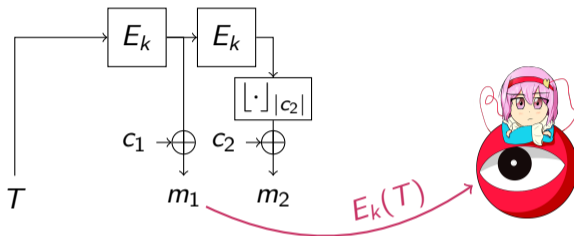
SIV structure



Release of Unverified Plaintext (RUP) Security [Andreeva et al., 2014]

Is the security compromised if an adversary has access to unverified decryption?

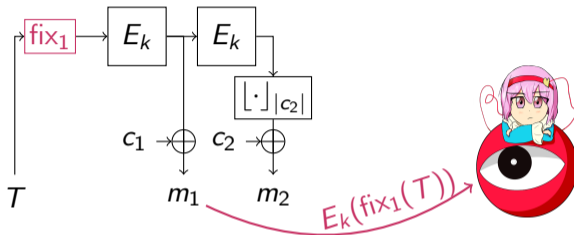
Simple RUP Attack on SUNDAE



Universal Forgery

Direct access to E_k : we can reconstruct the tag of any message.

Simple RUP Attack on SUNDAE **FIXED**



Universal Forgery

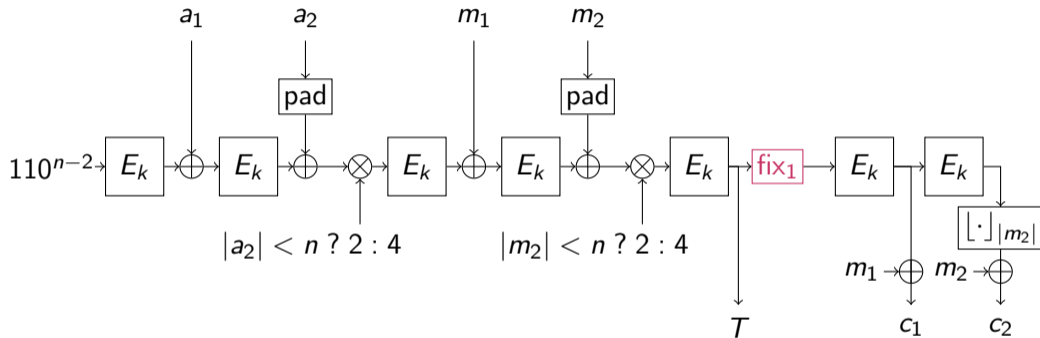
Direct access to E_k : we can reconstruct the tag of any message.

Denying the attack

Use a $\text{fix}_1(\cdot)$ function: fix the LSB to 1.

The adversary cannot simulate the authentication process anymore.

MONDAE



Result

$\text{fix}_1(\cdot)$ function: fix the LSB to 1.

MONDAE is provably RUP secure as well as AE secure.

Organisation

Type	Scheme	Strategy
Encryption	Counter Mode (CTR)	Missing Difference
Block Cipher	2-round Even-Mansour	3-XOR
Authentication	Double-bloc Hash-then-Sum MACs	4-XOR
Authenticated Encryption	SUNDAE	RUP attack

Conclusion

This thesis explored many aspects of provably secure symmetric schemes:

- We improved **computational cryptanalysis** for CTR (plaintext recovery), GMAC, SUM-ECBC, 2EM;
- We closed **Information Theoretic Gaps** for DbHtS MACs (later proof), XHX2 (instance of tweakable FX);
- We increased the **robustness** of SUNDAE at no cost with MONDAE.

Cryptanalysis and Proofs complete and even **inspire each-other** to fully assess the information theoretic security of a provable scheme.

New Directions

- The Computational vs Information Theoretic gap may be explained by **fine-grained complexity theory**. (3-XOR for 2EM, 4-XOR for DbHtS)
- Proofs in the ideal model are **usually** fine but it doesn't allow for a clear **separation** between the security of the scheme and the security of its primitive.