Provable Security of Submissions to the CAESAR Cryptographic Competition

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Telematics

submitted to

Graz University of Technology

Supervisor

Dipl.-Ing. Dr.techn. Florian Mendel

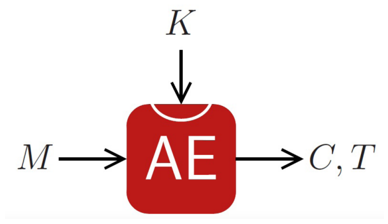
Institute for Applied Information Processing and Communications

Faculty of Computer Science

Graz, May 2015

MASTER'S THESIS

Provable Security of Submissions to the CAESAR Cryptographic Competition



GRAZ UNIVERSITY OF TECHNOLOGY

Author:
Robin ANKELE
`robin.ankele@student.tugraz.at`

Supervisors:
Bart MENNINK, KU Leuven
Elena ANDREEVA, KU Leuven
Assessor:
Florian MENDEL, TU Graz

Leuven, May 2015

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz,	______________________	______________________
	Date	Signature

Eidesstattliche Erklärung[1]

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am	______________________	______________________
	Datum	Unterschrift

[1]Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Acknowledgments

This thesis is the result of the contribution of many people. Therefore, with this preface I would like to take the opportunity to thank those, that supported me during my studies and to accomplish this thesis.

First of all, I would like to thank my advisors at the Katholieke Universitaet Leuven – Bart Mennink and Elena Andreeva. Thanks for being a constant source of guidance, and introducing me to the fairly complex world of provable security. Secondly, I would like to express my gratitude to my advisor at Graz University of Technology – Florian Mendel, for enabling me to perform my master's thesis and the supervision of my master's project.

Additionally, I would like to thank the promoters of my thesis – Bart Preneel and Vincent Rijmen – for inviting me to Leuven, to perform my thesis at COSIC. Special thanks to Péla Noé, for helping me with boring paperwork and other stuff.

Nevertheless, all this work could not be done without the constant support of my family and my friends. I am indebted to my twin brother Ralph, for an early review of this thesis and other useful comments to improve this thesis. It is my pleasure to thank all my friends at my hometown, and especially all my new friends, that I made during the stay in Leuven – for frequent distractions and making my stay most enjoyable.

Abstract

Modern cryptography is build upon the approach to design and develop provable secure primitives. Hereby, a security proof is typically the reduction from a problem $\mathcal{P}$ - related with the security of the candidate scheme, to a well known problem $\mathcal{P}_1$ of *e.g.* a random oracle or a one-way function.

An authenticated encryption scheme (AE), is a symmetric primitive that provides both – *data privacy* and *data integrity*. To identify a secure and robust authenticated encryption scheme, the CAESAR competition was recently announced. *COPA* is an AE composition scheme used in several CAESAR candidates.

We consider *COPA* in the nonce-respecting setting, and derive a variant of the scheme $\overline{COPA}$, which we prove secure beyond the birthday bound. For the proof, we use Patarin's coefficient-H technique – a well known technique for probable security of symmetric primitives. Furthermore, we show the applicability of our results to the CAESAR candidates AES-*COPA* and PRØST, where we increase the IND-CPA security bound from 2^{64} to 2^{83} encryption calls for AES-*COPA* and from $2^{64}(2^{128})$ to $2^{83}(2^{168})$ encryption calls for PRØST-*COPA*, respectively.

Moreover, we studied the mode *COPA* in the CAESAR candidate schemes Deoxys, Joltik and KIASU. Since *COPA* is used there in a slightly derivate version, we established another separate proof for *privacy* and *integrity*. Therefore, we can give first results on the conjecture of Jean *et al.*, that Deoxys, Joltik and KIASU achieve security beyond the birthday bound in the nonce-respecting setting.

Keywords. Provable Security, CAESAR, Authenticated Encryption, *COPA*, IND-CPA, INT-CTXT, Nonce, AES-*COPA*, PRØST, Deoxys, Joltik, KIASU

Contents

List of Figures

List of Tables

List of Algorithms

Notation

Notation and Terminology

$\mathcal{M}$	Message Space
$\mathcal{C}$	Ciphertext Space
$\mathcal{K}$	Key Space
$\mathcal{AD}$	Associated Data Space
$\mathcal{T}$	Tag Space
$\mathcal{IV}$	Initialization Vector Space
$\mathcal{N}$	Nonce Space
AE	Authenticated Encryption
AEAD	Authenticated Encryption with Associated Data
Pr	Probability
n	Block Size
ν	View - A set of input/output values

List of Mathematical Symbols

$a \oplus b$	Exclusive-or (XOR)	$a\|\|b$	Concatenation of a and b
$a \in A$	Element of set A	$\|a\|$	Length of a
a_i	*i*-th element	$a[i]$	*i*-th element
$a_{i,j}$	*j*-th value of *i*-th element	ϵ	Empty set
$a \ll b$	Left shift by *b* bits	$a \gg b$	Right shift by *b* bits

1 Introduction

Confidential messages that shall be transmitted via an insecure channel usually not only require *confidentiality* – such that only permitted entities can decrypt these messages, but also *authenticity* of their respective sender, in order to detect willful corruption of those messages. Such cases occur frequently in the real world – for example if an entity A wants to send an information M to an entity B, then we want *data privacy* to ensure that A's information remains confidential and additionally, we want *integrity* and *authenticity* to ensure the information was send by entity A and not modified during the transit.

Concerned with that problem, one could use an encryption scheme to transmit information M via the insecure channel to entity B. However, consider the following problem. $A's$ message contains the information `Transfer $100 to bank account X`. Now, a malicious adversary E can't read the encrypted message, but it maybe just flips some bits to alter the message to `Transfer $500 to bank account X`. The problem we are facing is that an encryption scheme only provides *privacy* and no *authenticity*. Moreover, an authentication scheme provides only *authenticity*, but no *privacy*. If we want a secure data transmission, we need a scheme that provides both goals, simultaneously.

An authenticated encryption scheme provides a secure and robust method to fulfill both requirements. Syntactically, an authenticated encryption scheme ($\mathcal{AE}$) is just a symmetric encryption scheme ($\mathcal{SE}$) that usually combines a privacy-only scheme with an authentication scheme. Authenticated encryption schemes can be divided into *one-pass* or *two-pass* schemes, where the former achieves AE by implementing a dedicated design and the latter combines a $\mathcal{SE}$-scheme with an authentication scheme ($\mathcal{F}$). A two-pass scheme is often realized by the generic composition paradigm introduced by Bellare *et al.* in [BN00].

The framework of provable security defines security notions, that can be applied to the primitive being analyzed, to describe the security level achieved by the primitive. Then, a security proof guarantees that there is no computationally limited adversary, that can perform these kind of attacks on the primitive with a non-negligible probability. Modern cryptography is build upon these design approach to guarantee security.

A privacy-only $\mathcal{SE}$ scheme is therefore secure if it achieves the notion of IND-CPA or IND-CCA – indistinguishabillity under a chosen plaintext or ciphertext attack. Moreover, an authentication scheme $\mathcal{F}$ is secure if it achieves the notion of INT-PTXT or INT-CTXT – the integrity of plaintext or ciphertext. Now, an $\mathcal{AE}$-scheme is secure if it achieves IND-CCA, where IND-CPA + INT-CTXT $\Rightarrow$ IND-CCA. In addition, an $\mathcal{AE}$-scheme is robust if it achieves the INT-RUP notion by Andreeva *et al.* defined in [ABL+14a] – to be secure under the release of unverified plaintext. More recently, especially many candidates of the CAESAR competition specify online authenticated encryption (OAE) – where the i^{th} output bit only depends on the input bits from 0 to i. Withal, the use of a nonce (*i.e.* number only used once) plays a huge role in AE and the related security notions and proofs – where we distinguish between nonce-ignoring and nonce-respecting adversaries.

Application of AE. Authenticated encryption is used in several networking protocols of the internet protocol suite. There, it is used to encrypt and authenticate data that is transmitted over a potentially insecure channel – to provide *data privacy* and *data integrity*. In more detail, the *Encrypt-then-Mac* approach is used in the protocol IPsec [Ken05], *Encrypt-and-Mac* is used in SSH [Ylo06] and *Mac-then-Encrypt* is used in SSL/TLS [Die08]. Moreover, NIST has standardized the AE-schemes GCM and CCM – which are used in the above mentioned protocols.

Existing Designs and Standards. Before the CAESAR competition, there exists several authenticated encryption modes – CWC [KVW04], CCM [Dwo04], EAX [BRW04], IAPM [Jut00], OCB [RK14], GCM [MV04], which are discussed in more detail in Section 4.3.1. Some of the mentioned AE-schemes are standardized in ISO/IEC 19772:2009.

CAESAR Competition. CAESAR was announced to establish a portfolio of secure and robust authenticated encryption schemes, which are available for widespread adoption. The competition should empathize the general cryptographic community, to gain more insight and knowledge about the design and analysis of authenticated encryption schemes. Moreover, the competition can be motivated by several recent attacks on TLS and Open SSL like BEAST [DR11], CRIME [Kel02], POODLE [DMK14] or Heartbleed [DKA$^+$14]. Additionally, Namprempre *et al.* [NRS14] show that sometimes also standards can be written wrong and offer some vulnerabilities. At the time this thesis was written, we stand between the first and second of four rounds until the final portfolio will be established.

Related Work & Our Contributions. *COPA* [ABL$^+$13] was proposed at Asiacrypt 2013 as the first fully parallelizable online authenticated encryption mode. Since then, *COPA* has been used in several CAESAR submissions to achieve AE. *COPA* is supported with a security proof given in the design document [ABL$^+$13]. Nevertheless, before the announcement of the CAESAR competition, to our knowledge, no cryptanalysis nor variants of *COPA* were published. In light of the CAESAR competition Ju proposed a forgery attack on AES-Copa [Lu15]. To our knowledge, no other attack on constructions using *COPA* have been published. Most recently, Nandi [Nan15] published an attack on *COPA* with non-multiple blocksize, reducing the *integrity* security claim of *COPA* to $2^{n/3}$ queries. However, the approach to construct a variant of *COPA* to achieve security beyond the birthday attack is new. 33 out of the 48 remaining CAESAR candidates are supported by a security proof. We aim to increase this number.

Our contribution, within this thesis, comprises the establishment of a security proof for a variant of the AE composition scheme *COPA* designed by Andreeva *et al.* [ABL+13]. *COPA* finds adaption in several CAESAR candidates – AES-Copa [ABL+14b], Prøst [KLL+14], Deoxys [JNP14b], Joltik [JNP14c] and KIASU [JNP14d]. Our variant, further called $\overline{COPA}$, makes use of a nonce and limits the adversary to be nonce-respecting. In this setting, we show that $\overline{COPA}$ achieves security beyond the birthday bound. Furthermore, we analyze the application of $\overline{COPA}$ to the above mentioned CAESAR candidates. Finally, we established a second proof for Deoxys, Joltik and KIASU using the mode *COPA* – in a slightly different setting.

Outline. This thesis is organized as follows. In Chapter 2, we introduce some basic mathematical concepts and cryptographic primitives used throughout the thesis.

Chapter 3 gives an overview on symmetric cryptography. First, we introduce some attack models, secondly we define symmetric primitives for encryption and authentication and finally we give an overview on cryptanalytic attacks on symmetric primitives.

Moreover, in Chapter 4 we define authenticated encryption. We illustrate different types of AE-modes like the generic composition paradigm and give and overview of the candidates from the CAESAR competition.

The concepts of provable security are introduced in Chapter 5. We discuss some security notions and give an overview of the most practical proving techniques.

The main parts of this thesis are the proofs of $\overline{COPA}$ and its application to the CAESAR candidates AES-Copa, Prøst, Deoxys, Joltik and KIASU given in Chapter 6. First, we denote some online AE notions and give three recently proposed attacks on OAE. Next, we describe the scheme *COPA* and give an analysis on how to improve this mode. We perform a *privacy* proof on $\overline{COPA}$ and give its application to AES-Copa, Prøst. Finally, we introduce a second proof for the candidates Deoxys, Joltik and KIASU.

We conclude this thesis in Chapter 7 and give an outlook on further analysis.

2 Mathematical Background

In this chapter, we introduce some mathematical basics used throughout this thesis. Moreover, we present some cryptographic preliminaries. Readers that are familiar with basic mathematics and cryptography can skip to Chapter 3.

2.1. Probability Theory

A *random variable* inherits different values due to change (*i.e.* randomness), where each value is assigned to a certain probability. The mathematical function to describe a random variable with its associated probabilities is the *probability distribution*.
An event **E** indicates the possible outcomes of an experiment. An elementary event expresses exactly one outcome. Each event has a probability assigned to it. A set has a *uniform distribution* if the probability for each element is the same. We write

$$a \overset{\$}{\leftarrow} A \tag{2.1}$$

when we choose a value a uniformly at random from a set A.

The *probability* of an event **A** is the likeliness that the event **A** occurs. The probability is any value bound between $Pr(\mathbf{A}) \in [0,1]$. Hence, the probability of event **A** is

$$Pr(A) = \frac{|\mathbf{A}|}{|\mathbf{\Omega}|} = \frac{\#\, favorable\ events}{\#\, all\ events}. \tag{2.2}$$

The probability that a single event **A** occurs is also called *marginal* probability. The complementary probability of an event **A**

$$Pr^c = 1 - Pr(\mathbf{A}) \tag{2.3}$$

is the case in which the event A does not occur.

If we have two sets A and B we call the union of those two sets $A \cup B$, where the resulting value is in A or B. Furthermore, we call $A \cap B$ the intersection of the sets A and B, where the resulting value must be in A and B. An event **A** is independent (*i.e.* mutual exclusive) from an event **B** if the occurrence of the event **A** does not influence the occurrence of **B** and vice versa. Otherwise these events are dependent. In the literature independent events are often called *with replacement* and dependent are called *without replacement*, which just denotes if the sample space Ω (*i.e.* set of possible values) is reset after each experiment or not.

Yet, the probability of two events **A** or **B** is

$$Pr(\mathbf{A} \cup \mathbf{B}) = Pr(\mathbf{A}) + Pr(\mathbf{B}) \tag{2.4}$$

$$Pr(\mathbf{A} \cup \mathbf{B}) = Pr(\mathbf{A}) + Pr(\mathbf{B}) - Pr(\mathbf{A} \cap \mathbf{B}) \tag{2.5}$$

where $Pr(\mathbf{A} \cap \mathbf{B})$ denotes the joint probability and the events in (2.4) are mutual exclusive and in (2.5) not.

2.1.1. Joint Probability

The probability that two events **A** and **B** occur at the same time is given by

$$Pr(\mathbf{A} \cap \mathbf{B}) = Pr(\mathbf{A}) * Pr(\mathbf{B}) \tag{2.6}$$

$$Pr(\mathbf{A} \cap \mathbf{B}) = Pr(\mathbf{A}|\mathbf{B}) * Pr(\mathbf{B}) = Pr(\mathbf{B}|\mathbf{A}) * Pr(\mathbf{A}) \tag{2.7}$$

where $Pr(\mathbf{A}|\mathbf{B})$ denotes the conditional probability and the events in (2.6) are mutual exclusive and in (2.7) not.

2.1.2. Conditional Probability

The probability that event **A** occurs given that event **B** already occurred is

$$Pr(\mathbf{A}|\mathbf{B}) = \frac{Pr(\mathbf{A} \cap \mathbf{B})}{Pr(\mathbf{B})} = \frac{Pr(\mathbf{B}|\mathbf{A}) * Pr(\mathbf{A})}{Pr(\mathbf{B})} \tag{2.8}$$

where the relation between $Pr(\mathbf{A}|\mathbf{B})$ and $Pr(\mathbf{B}|\mathbf{A})$ is given by the Bayes′ theorem.

2.1.3. Expected Value

The expected value of a random variable is the long-term average value of an experiment. In more detail, it is the probability-weighted sum of all possible values the random variable disposes. We denote the expected value of a random variable X as

$$E[X] = x_1 p_{x_1} + x_2 p_{x_2} + \cdots + x_n p_{x_n} \tag{2.9}$$

where for $1 \leq i \leq n$: x_i denotes the values X can attain, and p_{x_i} denotes with what probability x_i occurs. Additionally, the expected value holds the property of linearity. Thus, the expected value of two random variables X and Y is $E[X + Y] = E[X] + E[Y]$, even if X and Y are not statistically independent.

2.1.4. Markov's Inequality

Markov's inequality states an upper bound for the probability of a non-negative function of a random variable X that is greater or equal to a positive constant a. Then we can write the inequality as

$$Pr(X \geq a) \leq \frac{E[X]}{a}. \tag{2.10}$$

2.2. Cryptographic Preliminaries

In this section, we define and introduce some cryptographic concepts and basic primitives, respectively.

2.2.1. Cryptographic Concepts

We introduce the concepts of a nonce, initialization vector and a tweak.

Definition 1. *(Nonce): A nonce is a unique value. It can be either a counter (until it overflows) or a random value (need to store previous outcomes to detect repetitions). When generating a nonce, one must ensure to never repeat itself. Nonces can be used as IV.*

Definition 2. *(IV): An initialization vector is a fixed-size input to a cryptographic primitive that can be random or any arbitrary number, depending on the cryptographic primitive. An IV is publicly known (i.e. not secret).*

Definition 3. *(Tweak): A tweak can be any arbitrary number, secret or public known, which serves as an additional input to a cryptographic primitive.*

2.2.2. Cryptographic Primitives

In the following, we define some of the most basic cryptographic primitives which serve as building blocks for higher level (or more complicated) cryptographic primitives.

Definition 4. *(Random Oracle): A random oracle [BR93] is a map from $R : \{0,1\}^* \to \{0,1\}^*$, such that for every x, the bits of $R(x)$ are selected uniformly and independently.*

Definition 5. *(One-Way Function): A one-way function [MVO96] is a map from $f : \{0,1\}^* \to \{0,1\}^*$, such that for every x in the domain of f, it is easy to compute $f(x)$, but for essentially every y in the range of f, it shall be computationally infeasible (i.e. with negligible probability) to find a x such that $y = f(x)$.*

Pseudo Random Functions

Definition 6. *(Random Function): A random function [MVO96] is a map from $f : \{0,1\}^n \rightarrow \{0,1\}^n$, such that for every argument x in the domain, $f(x)$ gets set with random and independent values from the range of f.*

Definition 7. *(Pseudo Random Function): A pseudo-random function [KL07] is a efficient, length-preserving, keyed map from $f : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$, such that a PPT distinguisher (Def. 11) can distinguish between f and a random function, with only a negligible probability.*

Pseudo Random Permutations

Definition 8. *(Random Permutation): A random permutation [KL07] is a bijection (i.e. one-to-one onto map) from $p : \{0,1\}^n \rightarrow \{0,1\}^n$, such that for every argument x in the domain, $p(x)$ defines a random ordering of x.*

Definition 9. *(Pseudo Random Permutation): A pseudo-random permutation [KL07] is a efficient, keyed map from $p : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$, such that a PPT distinguisher (Def. 11) can distinguish between p and a random permutation, with only a negligible probability.*

2.3. Group and Field Theory

A group $\langle G, \bullet \rangle$ is a set G together with an operation $\bullet$. It must fulfill the following conditions:

◇ **Closure:** $\forall (a,b) \in G$ also the resulting value $(a \bullet b) \in G$
◇ **Associativity:** $\forall (a,b,c) \in G$ it must hold that $(a \bullet b) \bullet c = a \bullet (b \bullet c)$
◇ **Identity Element:** $\exists i \in G$ such that $\forall a \in G : a \bullet i = i \bullet a = a$
◇ **Inverse Element:** $\forall a \in G$ there $\exists b = a^{-1} \in G$ such that $a \bullet b = b \bullet a = i$

A field F is defined as a group with some additional requirements. Defined on the two operations (addition $+$, multiplication $\cdot$) it requires closure, associativity, existence of an identity and an inverse element and additionally defines:

- ◇ **Commutativity:** $\forall(a, b) \in F$ it must hold that $a + b = b + a$ and $a \cdot b = b \cdot a$
- ◇ **Distributivity:** $\forall(a, b, c) \in F$ it must hold that $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

GF(2^n). A Galois Field or finite field is a field with a finite number of elements (*i.e.* order of the group is 2^n) where each field element is evaluated modulo n. A point in the $GF(2^n)$ can be represented interchangeably as: (i) an abstract point in the field, (ii) an integer between $0..2^n - 1$, (iii) a n-bit string $a_{n-1} \dots a_1 a_0 \in \{0, 1\}^n$ or (iv) a formal polynomial $a(x) = a_{n-1}x^{n-1} + \dots + a_1 x + a_0$ with binary coefficients.

Addition in GF(2^n). To add two points a and b in the $GF(2^n)$ we just use the bitwise xor (*i.e.* $a \oplus b$).

Multiplication in GF(2^n). To multiply two points a and b in the $GF(2^n)$ we need to select first a irreducible polynomial $p(x)$ of degree n having binary coefficients. Then, we regard both points a and b as polynomials and form their product $c(x) = a(x) * b(x)$. Finally, we need to reduce $c(x)$ using $p(x)$ by evaluating $c(x)$ mod $p(x)$.

A multiplication by two is computationally simple. If the first bit of a is zero, then we just need to left-shift a by one bit (*i.e.* $2a = a \ll 1$). If the first bit is one, then we additionally need to add x^n to the resulting $a \ll 1$. Moreover, it is quite easy to multiply other small constants (*e.g.* $3a = 2a \oplus a$, $5a = 2(2a) \oplus a$ or $7a = 2(2a) \oplus 2a \oplus a$).

2.4. Advantage

We define Σ to be a finite alphabet (*e.g.* $\Sigma = \{0,1\}$). Moreover, let $\Sigma^\star$ be the set of all strings that result from arbitrary joining elements from Σ. A computational step would be some input $x \in \Sigma^\star$ transforming to an output $y \in \Sigma^\star$. Then, an algorithm $\mathcal{A}$ is a sequence of computational steps.

An adversary $\mathcal{A}$ is an algorithm, which transforms an input to an output. We denote by $\mathcal{A}^{\mathcal{O}}$ the access of an adversary $\mathcal{A}$ to an oracle $\mathcal{O}$, where $\mathcal{O}$ is a black box, that performs any computational steps. Likewise, we consider algorithms as Turing machines. Now, an algorithm is called *efficient* and runs in polynomial time if it fulfills the condition in Definition 10 and 11.

Definition 10. *(Polynomial Time). Let $\mathcal{A}$ be an algorithm, then there exists a $n_0 \in \mathbb{N}$, $c \in \mathbb{R}$ and a polynomial $p(n)$, where the time complexity of $\mathcal{A}$ is $T_{\mathcal{A}}(n)$ such that $\forall n \leq n_0 : T_{\mathcal{A}}(n) \leq c \cdot p(n)$.*

Definition 11. *(PPT algorithm): A probabilistic polynomial-time algorithm is an algorithm running in polynomial time that makes random coin tosses upon execution and terminates for some input x after $|p(x)|$ steps, for some polynomial p.*

Until noted otherwise, we assume all adversaries $\mathcal{A}$ and distinguishers $\mathcal{D}$ to be PPT algorithms. The distinguishing-advantage of a $\mathcal{D}$ is given by

$$\mathbf{Adv}(\mathcal{D}) = \left| Pr[\mathcal{D}^{\mathcal{O}_1} \Rightarrow 1] - Pr[\mathcal{D}^{\mathcal{O}_2} \Rightarrow 1] \right| \tag{2.11}$$

where the distinguisher $\mathcal{D}$ has either access to oracle $\mathcal{O}_1$ or $\mathcal{O}_2$ and outputs one or zero. The job of $\mathcal{D}$ is now to distinguish between these two oracles – using only the outcome of either $\mathcal{O}_1$ or $\mathcal{O}_2$. The advantage is normally bounded by several parameters like the time t, the number of queries q to the oracles, the length ℓ of those queries and the total length σ of the submitted data. Moreover, the advantage can also be defined as the statistical distance (*i.e.* total variation distance) between two probability distributions (*e.g.* X, Y). Then, for a deterministic distinguisher $\mathcal{D}$ the distinguishing-advantage is given by

$$\mathbf{Adv}(\mathcal{D}) = \delta(X, Y) = \frac{1}{2} \sum_x \left| X(x) - Y(x) \right| \tag{2.12}$$

An adversary/distinguisher is successful if its advantage is non-negligible. Hereby, a function $\epsilon(\cdot)$ is called *negligible* if for every polynomial $p(\cdot)$ there is an integer $n_0 \in \mathbb{N}$ such that

$$\forall n \geq n_0 : \epsilon(n) < \frac{1}{p(n)} \tag{2.13}$$

Moreover, this means that $\epsilon(n)$ must be exponentially small for $\forall n \geq n_0$.

3 Symmetric Cryptography

In this chapter, we give a short introduction to cryptography, mainly focused on symmetric cryptography. We describe the most notable primitives for encryption and authentication and end with some methods to analyze symmetric primitives.

3.1. Preliminaries

Cryptography comes from the greek *kryptós* (*i.e.* hidden) and *graphein* (*i.e.* writing) and is the science of secure communication in the presence of an attacker. Therefore, the main goals of cryptography are

◇ **Confidentiality** or **Privacy.** Only authorized entities can read messages.
◇ **Integrity.** Detection of change in data due to an adversary or data corruption.
◇ **Authentication.** Confirmation of the identity of an entity.
◇ **Non-Repudiation.** Undeniable involvement in an action (*e.g.* signature).

3.1.1. Cryptographic Security

In order to measure cryptographic security we categorize the security of a cryptosystem into one of the following categories.

Information Theoretic Security

A cryptosystem achieves information theoretic security if it can not be broken even if the adversary disposes of unlimited computational power. This is because the adversary simply doesn't possess enough information to be able to break the system which implies that such a system is cryptanalytically unbreakable. A special case of information theoretic security is the term introduced by Shannon, *perfect secrecy*.
An encryption scheme is perfectly secure, if and only if a ciphertext leaks no information about the plaintext, except the length of the message. The Vernam cipher or OTP (one-time pad) provides perfect secrecy.

$$C = M \oplus K \tag{3.1}$$

Equation (3.1): One-time pad. The ciphertext is just a simple xor of the message and key. It must hold that $|M| = |K|$. Furthermore, the key K is not allowed to be used twice and must be perfectly random.

Complexity Theoretic Security

Complexity theoretic security builds upon complexity theory. Therefore, the notion of Goldwasser and Micali about *semantic security* states: A cryptosystem is semantically secure if any probabilistic polynomial time algorithm given a certain ciphertext and the length of its message can not determine any partial information about its plaintext with a non-negligible probability higher than any other probabilistic polynomial time algorithm, given only the length of the message. From another viewpoint this means, that the knowledge of the length and any ciphertext (of an unknown message) doesn't reveal any partial information that can be feasibly extracted. Provable security builds upon these statements.

Cryptanalytic Security
The classical security approach is to build a cryptosystem, perform cryptanalysis and if any flaw is found to go back to step one. In this case cryptanalytic security means that a cryptosystems is secure against state-of-the-art cryptanalytic attacks. Modern cryptography aims to achieve security defined by the above mentioned categories.

3.1.2. Attack Models

An attack on a cryptosystem can be classified into the following attack models, depending on the information an adversary can access. We start with the weakest attack and continue with stronger attack models.

Known Ciphertext Attack. In this setting, an adversary has access only to the ciphertext and no access to the plaintext. It is the most likely case in any real world cryptanalysis, where the adversary tries to determine the plaintext from the ciphertext. In order to succeed with such an attack the adversary has usually some information about the plaintext (*e.g.* distribution, format, language).

Known Plaintext Attack. In this attack, an adversary has access to a limited number of plaintexts and their dedicated ciphertexts. An adversary tries to decrypt following ciphertexts or determine the secret key, using the known pairs.

Chosen Plaintext Attack. In this attack, an adversary can choose an arbitrary number of plaintexts which she submits to an encryption oracle and receives it ciphertexts. This is a very strong attack, because the adversary can choose any plaintext message she wants.

Moreover, in an *adaptive-chosen plaintext attack* the adversary is able to submit any sequence of plaintext and after each step she has the ability to analyze the ciphertext before choosing the next plaintext.

Chosen Ciphertext Attack. In this setting, an adversary has access to both plaintext and ciphertext and access to an encryption and decryption oracle. This is the strongest kind of attack and only a few ciphers are secure against such an attack. Moreover, we can refine this setting in the following ways.

◇ **CCA1.** A CCA1 attack is also called *lunchtime attack*, where an adversary can make adaptive queries up to a certain point in time (*e.g.* during the lunchtime of an user).

◇ **CCA2.** In this setting the adversary is allowed to submit any sequence of ciphertexts and after each step, before submitting the next ciphertext to observe and analyze its given plaintext.

Forgery. A forgery is the successful attempt that an adversary generates a *tag* τ to a message m such that a verification algorithm $Verify(m, \tau)$ outputs *true* for the case that the tag τ was not previously generated by a legitimate tag generation algorithm $Tag(m) = \tau$.

3.2. Encryption Schemes

An encryption scheme is a cryptographic construction that takes a message (*i.e.* plaintext) M and a key K as input and outputs a ciphertext C. More formally,

$$E : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^*$$

where E is a function, key K is a non-empty set from $\{0,1\}^k$. The message and ciphertext are of arbitrary length, but finite. We often denote an encryption scheme as $E_K(M) = C$ or $E(K, M) = C$, where $M \in \mathcal{M}$, $K \in \mathcal{K}$ and $C \in \mathcal{C}$.

Encryption schemes can be classified into *asymmetric* and *symmetric* encryption schemes. In the former we have two keys, a public key k_{pub} and a private key k_{priv} and in the latter we have only one key, the secret key k_{sec}. In this thesis, we consider only symmetric-key encryption schemes.

A symmetric-key encryption scheme $\mathcal{SE}$ consists of three algorithms, one for key generation, encryption and decryption. We write $\mathcal{SE} = (Key, Enc, Dec)$ to denote such a triple.

Algorithm 3.1: (Symmetric Key Algorithm) *Key* returns randomly a key from the key space. Enc_K returns the encrypted message. Dec_K returns either the message or a symbol $\perp$.

```
Alg $Key$
    $K \xleftarrow{\$} \{0,1\}^k$
    return $K$

Alg $Enc_K(M)$
    return $E_K(M)$

Alg $Dec_K(C)$
    return $E_K^{-1}(C)$
```

For *correctness* it must hold that $\forall M \in \mathcal{M} : D_K(E_K(M)) = M$. Additionally, we require that $|\mathcal{E}(K, M)|$ is at least $|M|$ and thereby the ciphertext expansion $|\mathcal{E}(K, M)| - |M|$ is a constant $\omega \geq 0$. Ideally, an encryption scheme is build from a pseudo-random function and behaves like a random function (*i.e.* the ciphertext should be close to be uniformly random from the set $\{0, 1\}^*$). In case of a block cipher it should behave like a pseudo-random permutation. An encryption scheme diminishes in security if the message is small (*e.g.* yes/no) or gets repeated. Therefore, different types of encryption schemes has been proposed.
A *probabilistic* encryption scheme (pE) is provided with a key and a plaintext and by using internally generated randomness it outputs a ciphertext.

An *IV-based* encryption scheme (ivE) $E_K^{IV} : \mathcal{K} \times \mathcal{IV} \times \{0, 1\}^* \rightarrow \{0, 1\}^*$ takes as an additional input an initialization vector (IV), selected uniformly at random and prepends it to the ciphertext.

As a third option there is a *nonce-based* encryption scheme (nE) proposed by Rogaway [Rog04b]. It is syntactically the same to the ivE scheme, but takes instead of a IV a nonce N. It is expected for a nE scheme to be secure as long the nonce does not get repeated.
There are several ways for the construction of encryption schemes. Most common is to build them from a block or a stream cipher.

3.2.1. Block Ciphers

A block cipher operates on a fixed input size n – the block size. Therefore, the encryption scheme splits the message M into n-bit blocks (*e.g.* $M = M_1||M_2 \cdots M_\ell$, where $|M_i| = n$ for each $1 \leq i \leq \ell$). Moreover, a block cipher is a function $\mathscr{E} : \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$, where $\mathcal{K}$ is a finite non-empty set and the message and ciphertext is split into block of size n. Nevertheless, a block cipher is build from a permutation such that the encryption $\mathscr{E}$ represents a permutation π and the decryption $\mathscr{D}$ represents the inverse permutation π^{-1}, respectively.

Even and Mansour [EM93] proposed a scheme building from a randomly chosen permutation and a pre/post-whitening key. It is defined as

$$f(x \oplus K_1) \oplus K_2$$

Dunkelman *et al.* [DKS12] proved that this scheme attains the same security even if only one key K is used. The most common principle is the iterated approach where a round function f_R is used and applied several times on the same message (*i.e.* $M_i = f_{R_i}(M_{i-1})$). An important subclass of the iterated approach is the substitution permutation network, where for each round a substitution in form of an SBox and afterwards a permutation over all bit is applied. Another approach is the Feistel network, where the message is split into two pieces L, R and each round consists of the following two operations

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= L_i \oplus F(R_i, K_i) \end{aligned}$$

Luby and Rackoff [LR88] showed that if the F function is a PRF then one can achieve a pseudo-random permutation after 3 rounds, and a strong PRP after 4 rounds. Notable block ciphers are DES [FTC+77], AES [DR98] or Blowfish [Sch94].

Processing of Fractional Data

If the message is not a multiple of the block size n or less then one block, then it must be preprocessed to be able to get encrypted by the block cipher. We can divide encryption into length-preserving (*i.e.* encipherment) and length-extending encryption.

10^* Padding. This length-extending procedure just appends a single 1 and as many 0's as required to fill the last message block.

Ciphertext Stealing. This length-preserving approach works as follows:

1) Encrypt $E_K(M_{n-1}) = X_{n-1}$ and split X_{n-1} into a (HEAD || TAIL), where HEAD is of size of $|M_n|$. HEAD of E_{n-1} will be C_n.
2) Encrypt $E_K(M_n || \text{TAIL}) = C_{n-1}$.

Tag Splitting. This length-preserving approach was introduced by Fleischmann *et al.* [FFL12] and works as follows:

1) Calculate a message checksum (or something related) $\tau = \bigoplus_{i=1}^{n} M_i$ and split it into $\tau = \tau^{\alpha}||\tau^{\beta}$. $|\tau^{\alpha}| = n - |M_n|$.
2) Encrypt $E_K(M_n||\tau^{\alpha}) = C_n||T^{\alpha}$.
3) Encrypt $E_K(\tau) = T^{\beta}||Z$. Discard Z.
4) Tag T is $T = T^{\alpha}||T^{\beta}$.

XLS. eXtention by Latin Squares is an approach introduced by Ristenpart *et al.* [RR07] and uses multi-permutations to achieve length-preserving encryption. XLS is used in *COPA* for message with length greater than a multiple of the block size (*i.e.* $|M| > n$). However, XLS was recently broken by Nandi [Nan14], who also showed an attack on *COPA*, when XLS is used [Nan15].

3.2.2. Tweakable Block Ciphers

A tweakable block cipher was first introduced by Liskov *et al.* [LRW02]. Additionally, to a plaintext M and a key K it takes a third input - a tweak T. A tweakable block cipher provides a new permutation for each new tweak T or key K. More formally,

$$E : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \rightarrow \{0,1\}^n$$

where E is a function, K, T is from a finite, non-empty space, the key space $\mathcal{K}$ and the tweak space $\mathcal{T}$. We often write $E_K^T(M) = C$ or $E(K, T, M) = C$. For *correctness* we demand $\forall M \in \mathcal{M} : D_K^T(E_K^T(M))$. Moreover, $|E_K^T(M)|$ is at least $|M|$. Notable tweakable block ciphers are the Hasty Pudding cipher [Sch98], the XE/XEX construction by Rogaway [Rog04a] or Threefish [FLS+10].

3.2.3. Stream Ciphers

A stream cipher is a symmetric cryptographic primitive that takes a plaintext and a key stream to produce a ciphertext. Mostly, the pseudo-random key stream is just xored to the plaintext. The keystream is generated from a short secret key (seed) which is input to a LFSR (Linear Feedback Shift Register) or any other pseudo-random key generator. Stream ciphers can be classified into *synchronous* and *asynchronous* designs, where in the former the key stream is independent of the plaintext and ciphertext and in the latter the key stream depends on several ciphertext bits. Notable stream ciphers are RC4 [Riv87], Salsa20 [Ber08] or Scream [HCJ02].

3.3. Authentication Schemes

An authentication scheme is a cryptographic construction consisting of a triple of algorithms, a key generation algorithm *Key* that returns a key K uniformly at random from the key space $\mathcal{K}$, a tag generation algorithm *Tag* that takes a message M and the key K and produces a digest τ; and a verification algorithm $Verify$ that takes the same message M, key K, digest τ and outputs whether the digest is legitimate (regarding the message) or not.

Algorithm 3.2: (Authentication Algorithm) - *Tag* produces a digest from message M. $Verify$ verifies if τ is a valid tag and returns either $\top$ (*i.e.* success) or $\perp$ (*i.e.* failure).

```
Alg $Key$
  $K \xleftarrow{\$} \mathcal{K}$
  return $K$

Alg $Tag_K(M)$
  $\tau \xleftarrow{\$} T(K,M)$
  return $\tau$

Alg $Verify_K(M,\tau)$
  $b \leftarrow V(K,M,\tau)$;
  if $b = 1$ then return $\top$
  else return $\perp$
```

Thus, such a scheme aims to provides integrity and authentication. The tag algorithm *Tag* takes an arbitrary but finite length message $M \in \mathcal{M}$ and a key $K \in \mathcal{K}$ and by producing a short (t-bit) digest $\tau \in \{0,1\}^t$ it ensures that the message can not be altered, and the digest can not be produced by anybody without knowledge of the key K. More formally,

$$T : \mathcal{K} \times \{0,1\}^* \to \{0,1\}^t$$

The verification algorithm $\mathcal{V}$ takes the same inputs as the tag generation algorithm and in addition takes also the produced tag $\tau \in \{0,1\}^t$. It returns either $\top$ as a sign of a successful verification or a symbol $\perp$ for failure. More formally,

$$V : \mathcal{K} \times \{0,1\}^* \times \{0,1\}^t \to \{\perp, \top\}$$

For *correctness* we require that $\forall M \in \mathcal{M}, \tau \leftarrow Tag_K(M) : Verify_K(M, \tau) = \top$.

3.3.1. Hash Functions

A hash function $f(\cdot)$ is a cryptographic primitive mapping an arbitrary length message M to a fixed length digest τ. Ideally, it represents a one-way function. Hence, the digest $\tau = f(M)$ should be *efficient* computable for any $M \in \mathcal{M}$ but it should be infeasible to compute the inverse $M = f^{-1}(\tau)$. Moreover, it should be infeasible to change the message M without changing the digest τ or to find two different messages M_1, M_2 with the same digest $\tau_1 = \tau_2$. More formally,

$$f : \{0,1\}^* \to \{0,1\}^n$$

A hash function is used to provide integrity of a message M. Due to the recent SHA-3 competition we have a huge pool of well-analyzed cryptographic hash functions (*e.g.* Keccak [BDPVA11], Skein [FLS+10], Grøstl [GKM+11], . . .).

3.3.2. Message Authentication Codes

A message authentication code (`MAC`) is the most prominent way to implement an authentication scheme. It usually combines a hash function plus some additional pre/post-processing using a secret key K to produce a *mac* (*i.e.* tag, hash). Therefore, it provides integrity (using the hash function) and authenticity (due to the secret key). A `MAC` consists of the two algorithms, `Tag` and `Verify`. The value *mac* is normally appended to the message ($M||mac$) and send to the receiver. Notable message authentication codes are HMAC [CBK97], PMAC [BR02] and VMAC [Kor07].

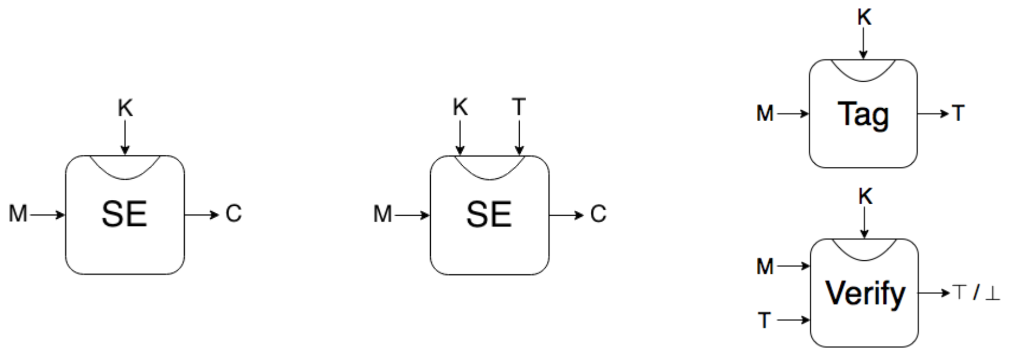


Figure 3.1.: Summary. (**left**) A symmetric encryption scheme takes M and K and outputs C. (**middle**) A tweakable encryption scheme takes additionally a tweak T as input. (**right**) A MAC consists of two algorithms – `Tag` takes M and K and produces T; `Verify` takes T and K and outputs either $\{\top, \bot\}$.

3.4. Modes of Operation

A mode of operation is an algorithm, using a block cipher, to provide confidentiality or even authenticity to an information system. A block cipher is per se not able to encrypt more than one block of data. Therefore, a mode of operation describes how to apply a block cipher to multiple blocks of data. Often such modes of operation require an initialization vector (`IV`), that has to be non-repeating and/or random. Furthermore, if the length of data is not a multiple of the block size of the underlying block cipher then the last block needs to be padded by some padding rule. The most known modes of operation are `ECB`, `CBC`, `CFB`, `OFB`, `CTR`, where the most prominent ones are described in more detail below. Some of the modes support parallelizable encryption and decryption or even some error correcting properties.

Electronic Codebook Mode (ECB). It divides the message into blocks and encrypts each block separately. Thus, it supports parallelizable encryption and decryption and one bit change in the plaintext/ciphertext influences only one block. More formally,

$$C_i = E_K(M_i)$$

for $1 \leq i \leq |M|/n$. Decryption is exactly the inverse of encryption. Nevertheless, a big disadvantage of ECB is that it produces identical ciphertext for identical plaintext blocks. Since no patterns are hidden it is not recommended to use this mode.

Ciphertext Blockchaining Mode (CBC). The CBC mode xors the previous ciphertext C_{i-1} to the current plaintext M_i prior to each encryption. To avoid identical ciphertext blocks for identical message blocks it uses a non-repeating, random IV for the first ciphertext block. In this mode only decryption is parallelizable. Only one bit change in a plaintext or IV changes all the following ciphertexts. More formally,

$$C_0 = IV, \quad C_i = E_K(P_i \oplus C_{i-1})$$

for encryption and for decryption

$$C_0 = IV, \quad P_i = D_K(C_i) \oplus C_{i-1}$$

for $1 \leq i \leq |M|/n$.

Counter Mode (CTR). In the CTR mode we turn the block cipher into a stream cipher. Thus, using the block cipher E_K and a counter ctr we produce a key stream, which is than xored to the plaintext message M. Furthermore, a nonce N is used to provide a unique counter for each block where $(nonce||ctr)$ is inputed to each block. CTR provides parallelizable encryption and decryption. More formally,

$$C_i = P_i \oplus E_K(N||ctr)$$

for encryption and for decryption

$$P_i = C_i \oplus E_K(N||ctr)$$

for $1 \leq i \leq |M|/n$. One bit change in the plaintext/ciphertext influences only one bit in resulting output.

3.5. Analysis of Symmetric Key Primitives

In this section, we give a short description of some attacks on the previously introduced symmetric key primitives. In order to prove a scheme secure one should have knowledge of the most basic state-of-the-art attacks on the primitives used (*i.e.* to be able to protect against those attacks).

3.5.1. Generic Attacks

In a generic attack, we treat the symmetric key primitive as a *black box*. Then, we analyze the in- and output values of this black box and try to find some interesting behavior (*e.g.* non-uniform distribution of output values).

Brute Force. In a brute force attack an adversary tries all possible input values (by searching through the whole input space) until she hits the correct result. *Exhaustive key search* is one instance of a brute force attack, where the adversary tries all possible key combinations to obtain a message from a given ciphertext. The worst case runtime of such an attack is $\mathcal{O}(2^n)$ for some security parameter n (*e.g.* key-, hash-size). All primitives should be designed with a large enough security parameter n to mitigate those kind of attacks.

Distinguishing Attack. In a distinguishing attack an adversary tries to distinguish between an idealized model (*e.g.* random oracle) and the real world model of the cryptographic primitive. She succeeds if she can distinguish between those two worlds in less then brute force runtime (*i.e.* $\mathcal{O}(2^n)$).

Birthday Attack. A birthday attack exploits the mathematics behind the birthday problem. Hence, an adversary tries to find a collision between two different inputs x_1, x_2 to a function f (*i.e.* $f(x_1) = f(x_2)$). By randomly selecting each input x to the function f, where each output is drawn with the same probability (*i.e.* uniform), an adversary has $\binom{q}{2} \approx q^2/2$ possible combinations by submitting q queries (*e.g.* for $x_1, x_2 \overset{\$}{\leftarrow} \{0,1\}^n$ we have $Pr(x_1 = x_2) \approx 0.3$ for $q = 2^{n/2}$).

Time/Memory Tradeoff. In a time-memory tradeoff attack [Hel80] an adversary can reduce the execution time of the attack by increasing the memory requirements. Hence, she trades one parameter in favor of the other by using precomputed tables to reduce the execution time. Typically, such an attack consists of two phases – an *offline* phase (pre-computation) where she explores the primitive and precomputed values in large tables; and an *online* phase (real time) where she tries to attack the primitive using the precomputed values.

3.5.2. Hash Function Attacks

Hash functions are the underlying primitives of an authentication scheme. Consequently, a secure hash function should be indistinguishable from its idealized version – a one way function. To be secure, a hash function has to resist against the following type of attacks.

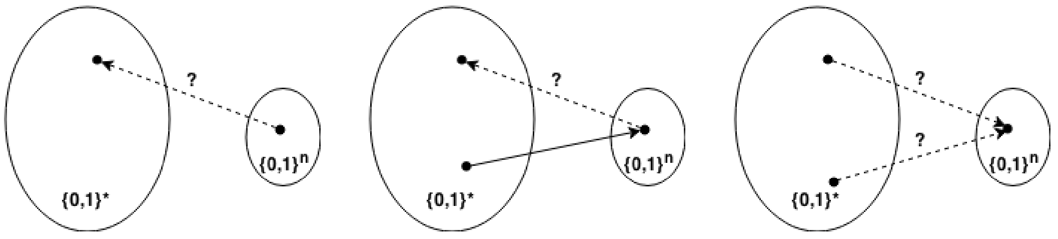


Figure 3.2.: Properties of a hash function. (**left**) Preimage resistance. (**middle**) 2nd-preimage resistance. (**right**) Collision resistance.

Preimage. Given a hash value $\tau = f(M)$ it should be computationally infeasible for any adversary $\mathcal{A}$ to find the message M. A hash function is considered preimage resistant if a preimage attack has a time complexity of $O(2^n)$.

Second Preimage. Given a message M_1 and its hash $\tau_1 = f(M_1)$ it should be computationally infeasible to find a second message $M_2 \neq M_1$ such that $f(M_1) = f(M_2)$. A hash function is considered 2^{nd} preimage resistant if a 2^{nd} preimage attack has a time complexity of $O(2^n)$.

Collision. It should be computationally infeasible to find two messages M_1 and M_2 such that $M_1 \neq M_2$ and $f(M_1) = f(M_2)$. A hash function is considered collision resistant if a collision attack has a time complexity of $O(2^{n/2})$.

3.5.3. Differential and Linear Cryptanalysis

In the following, we introduce differential and linear cryptanalysis one of the most widely used techniques to analyze block ciphers and hash functions.

Differential Cryptanalysis. This technique is based on a chosen-plaintext attack and was discovered by Eli Biham and Adi Shamir in the late 1980s. It studies the impact of differences in the input of a primitive with regards to differences in the output of the primitive (*i.e.* a differential is a relation between Δ_{in} and Δ_{out}). Primarily, a xor difference between two values is used, but any other difference can be used too. A differential characteristic denotes the path of a difference through several rounds of a cipher, that hold with a certain probability. Then, an adversary tries to find and connect several characteristics. Moreover, she needs to find an input value that follows these characteristics. The main goal of differential cryptanalysis is to exclude key-candidates that are not possible due to the propagation of the differences. Additionally, in differential cryptanalysis there exists some specialized attacks like impossible, truncated or high-order differential cryptanalysis.

Linear Cryptanalysis. This technique is based on a known-plaintext attack and was discovered by Mitsuru Matsui in 1992. Thereby, an adversary tries to find affine approximations of a primitive. Linear cryptanalysis consists of two parts. In the first step, an adversary tries to generate linear equations from plaintext, ciphertext and key bits that have a high bias (*i.e.* probability close to 0 or 1). The piling-up lemma [Mat94] is often used to construct linear approximations. In the second step, she tries to conjecture possible key bits by applying known plaintext-ciphertext pairs on the linear equations.

4 Authenticated Encryption

In this chapter, we introduce authenticated encryption (AE). AE is a symmetric encryption algorithm, which usually combines an encryption scheme and an authentication scheme to achieve confidentiality, integrity and authenticity simultaneously. In the following, we give a precise definition of AE, illustrate different types of AE and give an overview of candidates of the ongoing CAESAR competition.

4.1. Preliminaries

Loosely speaking, an authenticated encryption scheme is just a combination of a confidentiality and authentication mode. Moreover, the need for a secure authenticated encryption scheme arises from the observation that the combination of an encryption and an authentication mode is often difficult and error prone (*e.g.* implementation errors, nonce/IV restrictions, key restrictions, . . .).

AE. An AE scheme is a triple $\Pi = (Key, Enc, Dec)$ – a key generation algorithm *Key* that outputs a key K from the key space $\mathcal{K}$, an encryption algorithm *Enc* that takes a message M as an input and outputs a ciphertext $\mathscr{C}$, which is normally a combination of a ciphertext C and a tag T to achieve confidentiality, integrity and authenticity; and an decryption algorithm *Dec* that takes the ciphertext $\mathscr{C}$ as input and outputs either the message M or a rejection symbol $\perp$ in case of an error.

Algorithm 4.1: (Authenticated Encryption Algorithm) *Key* returns randomly a key from the key space. Enc_K returns the encrypted message. Dec_K returns either the message or a symbol $\perp$.

```
Alg $Key$
  $K \xleftarrow{\$} \{0,1\}^k$
  return $K$

Alg $Enc_K(M)$
  return $E_K(M)$

Alg $Dec_K(\mathscr{C})$
  return $D_K(\mathscr{C})$
```

More formally,

$$E : \mathcal{K} \times \{0,1\}^* \rightarrow \{0,1\}^*$$

and

$$D : \mathcal{K} \times \{0,1\}^* \rightarrow \{0,1\}^* \cup \{\perp\}$$

where (E, D) are functions, K is from a non-empty finite set and $M, \mathscr{C}$ is of arbitrary but finite length. We denote $E_K(M) = \mathscr{C}$ or $E(K, M) = \mathscr{C}$ where $\mathscr{C} = (C||T)$ as the encryption and $D_K(\mathscr{C}) = M/\perp$ or $D(K, \mathscr{C}) = M/\perp$ as the decryption. We often also write E_K^{-1} instead of D_K. For *correctness* we demand that $\forall M \in \mathcal{M}, E_K(M) = \mathscr{C} : D_K(\mathscr{C}) = M$. Moreover, $|E_K(M)|$ must be at least $|M|$. In order to provide authenticity the encryption must be length-increasing. Thus, the ciphertext expansion $|E_K(M)| - |M|$ is a constant $\tau > 0$.

AEAD. After analysis of existing AE schemes and emergence of new ones it was realized that in many real-world applications (such as network protocols) not all data needs to be encrypted – moreover it consists of secret and non-secret parts, where it would be nice if the secret parts are encrypted and both, the secret and non-secret parts get authenticated. From this, the notion of authenticated encryption with associated data (AEAD) was introduced, where the non-secret part is called *associated data*. AEAD is similar to AE, but takes additionally associated data as a third input. The associated data is normally some header information that only needs authentication and no privacy. More formally the notion of AEAD,

$$E : \mathcal{K} \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$$

and

$$D : \mathcal{K} \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \cup \{\perp\}$$

where we have the same parameters as in AE and additionally AD which is of arbitrary but finite length and from the associated data space $\mathcal{AD}$. We denote $E_K(AD, M) = \mathscr{C}$ or $E(K, AD, M) = \mathscr{C}$ where $\mathscr{C} = (C||T)$ as the encryption and $D_K(AD, \mathscr{C}) = M/\perp$ or $D(K, AD, \mathscr{C}) = M/\perp$ as the decryption. Again, for *correctness* we demand $\forall M \in \mathcal{M}, AD \in \mathcal{AD}, E_K(AD, M) = \mathscr{C} : D_K(AD, \mathscr{C}) = M$. Also, $|E_K(AD, M)|$ depends on $|AD|$ and $|M|$ and must be at least $|M|$.

nAE. Nonce-based authenticated encryption is a specific subclass of AEAD which takes as an additional security parameter a nonce. More formally,

$$E : \mathcal{K} \times \mathcal{N} \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$$

and

$$D : \mathcal{K} \times \mathcal{N} \times \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \cup \{\perp\}$$

with the same parameters as AEAD and additional a fixed-length nonce N from the finite and non-empty set $\mathcal{N}$. We denote $E_K(N, AD, M) = \mathscr{C}$ or $E(K, N, AD, M) = \mathscr{C}$ as the encryption and $D_K(N, AD, \mathscr{C}) = M/\perp$ or $D(K, N, AD, \mathscr{C}) = M/\perp$ as the decryption. For *correctness* we demand $\forall M \in \mathcal{M}, AD \in \mathcal{AD}, N \in \mathcal{N}, E_K(N, AD, M) = \mathscr{C} : D_K(N, AD, \mathscr{C}) = M$. $|E_K(N, AD, M)|$ depends on $|AD|$ and $|M|$ and must be at least $|M|$.

Properties of AE

In the following, we give some definitions related to AE schemes. Depending on the construction (generic composition or dedicated AE scheme) an AE scheme is either one pass or two pass.

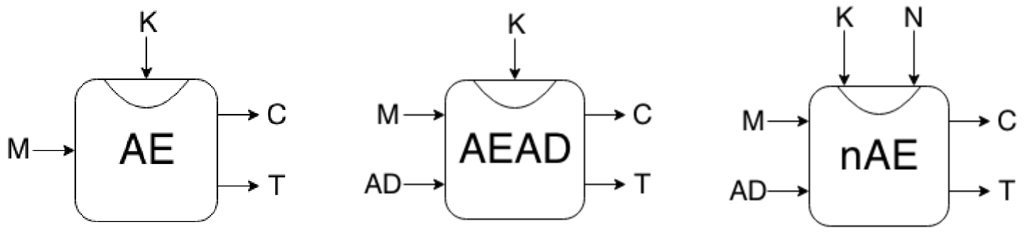


Figure 4.1.: Classification of AE. (**left**) Authenticated Encryption. (**middle**) Authenticated Encryption with Associated Data. (**right**) Nonce-based Authenticated Encryption with Associated Data.

Definition 12. *(One Pass/Two Pass): In a one-pass scheme the message (and associated data) is processed only once to produce a ciphertext and a tag value (i.e. to achieve privacy and authenticity), where in a two-pass scheme it is processed twice (e.g. in the first pass to produce a ciphertext and in the second pass to generate a tag or vice versa).*

For resource-restricted devices (*e.g.* RFID tags) with only a small buffer for message/ciphertext storage AE schemes can be based on an online cipher to mitigate these restrictions.

Definition 13. *(Online AE): Online authenticated encryption requires the encryption and/or decryption process to behave like an* online *function, whereas in an online function the output at position i depends only on the previously processed inputs (e.g.* $out_i = \bullet_{j=1}^{i} in_j$*, where* $\bullet$ *is any operation).*

Some AE schemes make use of nonces to achieve a higher security margin. Thus, there exists two worlds – a nonce-reusing and a nonce-respecting world where an adversary reuses or respects nonces, respectively.

Definition 14. *(Nonce Reuse): In the nonce-reuse setting an adversary* $\mathcal{A}$ *is allowed to reuse nonces to perform an attack.*

Definition 15. *(Nonce Respecting): In the nonce-respecting setting an adversary* $\mathcal{A}$ *is **not** permitted to reuse a nonce to perform an attack.*

Security and Robustness of AE. To achieve *security* and *robustness* an AE scheme must accomplish the following properties. We consider an AE

scheme secure if it provides data privacy (*i.e.* it is indistinguishable from an ideal authenticated encryption scheme – IND-CPA) and is secure against data forgery (*i.e.* it achieves ciphertext integrity – INT-CTXT). Moreover, we say an AE scheme $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ is secure iff the IND-CPA + INT-CTXT advantage is negligible for a nonce-respecting adversary $\mathcal{A}$.
We consider an AE scheme robust if it provides security against nonce-reusing adversaries and against the release of unverified plaintext. While nonces should be non-repeating and/or random in general, it is often hard to achieve these properties in the real world. This could be due to a reset of the device or a simple buffer overflow of a counter used as nonce. As mentioned above, resource-restricted devices often don't even have the possibility to provide the required resources to generate randomness or to detect repeats.

Definition 16. *(RUP): An AE-scheme is secure against the release of unverified plaintext if any adversary $\mathcal{A}$, that has access to oracles $\mathcal{E}_{\mathcal{K}}, \mathcal{D}_{\mathcal{K}}, \mathcal{V}_{\mathcal{K}}$ – an encryption, decryption and verification oracle, has only a negligible probability to produce a forgery.*

RUP is a notion by Andreeva *et al.* [ABL^{+}14a] and describes the problem of an online decryption function to release plaintext to a potential adversary $\mathcal{A}$ before verifying the tag value. This is due to the online property – to release plaintext as it comes from the decryption function. Moreover, a problem is when the buffer-size on the decryption side is too small to store the plaintext before receiving the tag.

Additionally, Rogaway *et al.* defined a notion called MRAE [RS06].

Definition 17. *(Nonce MRAE): An AE-scheme is called MRAE if it maintains security in case of nonce repetitions.*

In such a scheme the reuse of a nonce affects privacy in such a way, that an adversary $\mathcal{A}$ can detect repetitions of triples (N, AD, M) and authenticity remains. However, a problem with MRAE is that it can per definition not be online. Since for MRAE the ciphertext relies on every bit of input it can not be online, whereas in an online cipher the ciphertext relies on the currently processed input bits. Therefore, Fleischman *et al.* introduced a notion called OAE [FFL12], which provides security for an online AE scheme up to the longest common prefix (LCP) in case of a nonce-reuse.

4.2. Types of AE

There are several different types of authenticated encryption schemes. In the past, to construct an AE mode one just combined an encryption scheme and an authentication scheme. Bellare and Namprempre [BN00] defined some classification among AE schemes called *generic composition*. They defined three different construction schemes to combine a SE scheme and a MAC. Later on, dedicated AE schemes have gained popularity due to efficiency reasons (they perform a one-pass over the message) and parallelizability is only given in a one-pass scheme. Most recently, the CAESAR competition was announced, which introduced constructions from a broad architectural range. The majority of CAESAR candidates are block cipher or permutation based. Nevertheless, there are some stream cipher based constructions and some more exotic constructions like the compression function based OMD [CMN$^+$14].

4.2.1. Generic Composition

The generic composition paradigm was introduced by Bellare and Namprempre in 2000, where they showed that there are three ways to construct an $\mathcal{AE}$ mode by combining a $\mathcal{SE}$ scheme and a MAC. Furthermore, they argued that only *Encrypt-then-MAC* is generally secure. In 2013, Namprempre *et al.* [NRS14] showed that this argumentation holds only for probabilistic AE (by combining a pE + MAC), but is wrong for other constructions (*e.g.* nAE).

In the following, we introduce the three different composition schemes. Let the triple $\Pi = (Key, Enc, Dec)$ be an authenticated encryption scheme, with $\mathcal{SE} = (Key', Enc', Dec')$ an IND-CPA secure symmetric encryption scheme and $F : \{0,1\}^k \times \{0,1\}^* \rightarrow \{0,1\}^n$ a pseudo-random function. The key generation algorithm *Key* is the same for all three schemes.

```
**Alg** $Key$
    $K_e \leftarrow Key';\ K_m \xleftarrow{\$} \{0,1\}^k;$ **return** $K_e||K_m$
```

```
**Alg** $Enc_{K_e||K_M}(M)$
    $C' \leftarrow Enc'_{K_e}(M)$
    $T \leftarrow \mathcal{F}_{K_m}(M)$
    **return** $(C'||T)$
```

```
**Alg** $Dec_{K_e||K_M}(C'||T)$
    $M \leftarrow Dec'_{K_e}(C')$
    **if** $T = \mathcal{F}_{K_m}(M)$ **return** M
    **else return** $\perp$
```

Encrypt-and-Mac (E&M)
Encrypt-and-MAC is used in the secure shell (SSH) protocol. It is defined as follows:

Encrypt-and-MAC achieves the following security notions (see Section 5.2 for details).

Security	Achieved?	Arguments
IND-CPA	No	Detection of repeats.
INT-CTXT	No	Modification of C' s.t. $Dec'_{K_e}(C')$ is unchanged.

Mac-then-Encrypt (MtE)
Mac-then-Encrypt is used in the secure socket layer (SSL) and transport layer security (TLS) protocol. It is defined as follows:

```
**Alg** $Enc_{K_e||K_M}(M)$
    $T \leftarrow \mathcal{F}_{K_m}(M)$
    $C \leftarrow Enc'_{K_e}(M||T)$
    **return** $C$
```

```
**Alg** $Dec_{K_e||K_M}(C)$
    $M||T \leftarrow Dec'_{K_e}(C)$
    **if** $T = \mathcal{F}_{K_m}(M)$ **return** M
    **else return** $\perp$
```

Mac-then-Encrypt achieves the following security notions.

Security	Achieved?	Arguments
IND-CPA	Yes	None.
INT-CTXT	No	Modification of C s.t. $Dec'_{K_e}(C)$ is unchanged.

Encrypt-then-Mac (EtM)

Encrypt-then-Mac is used in the internet protocol security (IPsec) protocol. It is defined as follows:

```
**Alg** $Enc_{K_e||K_M}(M)$
    $C' \leftarrow Enc'_{K_e}(M)$
    $T \leftarrow \mathcal{F}_{K_m}(C')$
    **return** $C'||T$

**Alg** $Dec_{K_e||K_M}(C'||T)$
    $M \leftarrow Dec'_{K_e}(C')$
    **if** $T = \mathcal{F}_{K_m}(C')$ **return** M
    **else return** $\perp$
```

Encrypt-then-Mac achieves the following security notions.

Security	Achieved?	Arguments
IND-CPA	Yes	None.
INT-CTXT	Yes	None.

4.2.2. Block Cipher Based AE

The majority of CAESAR candidates are block cipher based and define an authenticated encryption mode by performing some pre/post-processing on each block iteration (*e.g.* using tweaks). Full round AES or some sub-rounds of AES are the mostly used instances of the block cipher. The main reasons to choose a block cipher as an underlying primitive is its efficiency (for AES there exist instruction set extensions (`AES-NI`) – for Intel and AMD processors), parallelizable implementations and block ciphers are well known and well-analyzed primitives due to the AES competition. While as a negative aspect, constructions using block ciphers (without nonces) diminish in security due to the birthday problem, which leads to larger block sizes.

4.2.3. Permutation Based AE

Another big fraction of CAESAR candidates is permutation based. Authenticated encryption is achieved either by using a sponge function (*e.g.* APE [ABB+15]) or by using dedicated permutations and some pre/post-processing. A mode using a dedicated permutation can be easily converted to a block cipher based mode by applying the Even Mansour scheme such as it is done in Prøst [KLL+14] or Minalpher [STA+14]. Authenticated encryption modes based on sponge functions benefit from the advantage that by construction they are easily build to be inverse-free and online which supports the usage of authenticated encryption in resource-constrained devices such as RFID tags. Furthermore, sponge functions achieve security up to $2^{c/2}$, with c the capacity part. Jovanovic *et al.* proved in their paper [JLM14] that this bound can be extended to $2^{b/2}$, where b is the permutation size.

4.3. CAESAR Competition

The Competition for Authenticated Encryption: Security, Applicability, Robustness [Ber] is an ongoing competition, which started in March 2014. The aim of the competition is to select a portfolio of secure and reliable AE schemes and for the cryptographic community to gain more insights into AE like in the previous SHA-3 and AES competition. The CAESAR call for submissions defines an AE scheme as follows.

An authenticated cipher defines five inputs and one output. Let $s, p, k \geq 1$, $M \in \{0,1\}^*$ denote a variable-length **plaintext**, $AD \in \{0,1\}^*$ denote a variable-length **associated data**, $SMN \in \{0,1\}^s$ denote a fixed-length **secret message number**, $PMN \in \{0,1\}^p$ denote a fixed-length **public message number** and $K \in \{0,1\}^k$ denote a fixed-length **key**. The output $C \in \{0,1\}^*$ is a variable-length ciphertext which provides privacy and integrity. An authenticated encryption scheme is a triple $\Pi = (Key, Enc, Dec)$ – a key-generation procedure, that returns a key choose uniformly at random from the key space $\mathcal{K}$, a deterministic encryption function $\mathcal{E}_K(M, AD, SMN, PMN)$ and its inverse a decryption function $\mathcal{D}_K(C, AD, PMN)$. Moreover, the signatures are defined as

$$\mathcal{E} : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^s \times \{0,1\}^p \times \{0,1\}^k \to \{0,1\}^*$$

$$\mathcal{D} : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^s \times \{0,1\}^p \times \{0,1\}^k \to \{0,1\}^*$$

For *correctness* we require that the message M and the secret message number SMN can be recovered from ciphertext C, associated data AD, public message number *PMN* and the key *K*. Nevertheless, it must hold that for every input from its input space that $\mathcal{D}_K(\mathcal{E}_K(M, AD, SMN, PMN), AD, PMN) = \{M, SMN\} \neq \perp$. The support of a *SMN* and a *PMN* is not needed and can therefore be 0-bits. There are different security requirements on each input value as given in Table 4.1.

Table 4.1.: Requirements for the input values of an AE scheme for the CAESAR competition

	Confidentiality	**Integrity**
Plaintext	yes	yes
Associated data	no	yes
Secret message number	yes	yes
Public message number	no	yes

The CAESAR call for submission stated that every candidate needs to state their advantage against the currently widely-used AES-GCM [MV04].

4.3.1. Current Authenticated Encryption Schemes

In the following, we give a short description of the current existing authenticated encryption schemes.

CWC. `CWC` [KVW04] combines the `CTR` mode for data encryption with an efficient polynomial Carter-Wegman MAC [CW79]. It is two-pass, parallelizable and provable secure.

CCM. `CCM` [Dw004] combines the `CTR` mode with `CBC`-MAC. It is two-pass, non-parallelizable, provable secure and used in IEEE 802.11i, IPsec and TLS.

EAX. `EAX` [BRW04] is a successor of `CCM` and combines `CTR` and `OMAC`. It has the same properties as `EAX`, in addition it supports pre-processing of AD and is online.

IAPM. `IAPM` [Jut00] was the first one-pass scheme. It is fully parallelizable and provable secure.

OCB. `OCB` [RK14] is the successor of `IAPM`. It is fully parallelizable and provable secure. `OCB` is also a CAESAR candidate [RK14].

GCM. `GCM` [MV04] is the successor of `CWC`. Instead of the expensive integer multiplications it operates in the binary galois field. It is fully parallelizable and provable secure. `GCM` is the current mostly used AE scheme and standardized by NIST. It is used in `IPsec`, `SSH` and `TLS` and included in NSA's Suite B for Cryptography.

4.3.2. First Round Candidates

There were 57 submissions as first round candidates. Table 4.2 lists the candidates, where highlighted in red are submission are withdrawn and highlighted in green are submissions that are considered in this thesis. In Section 4.3.3 we give a classification and some statistics about the first round candidates.

Table 4.2.: First Round Candidates

ACORN	++AE	AEGIS	AES-CMCC	🟥 AES-COBRA
🟩 AES-COPA	AES-CPFB	AES-JAMBU	AES-OTR	AEZ
AVALANCHE	Artemia	Ascon	CBA	🟥 CBEAM
CLOC	🟥 Calico	🟩 Deoxys	🟦 ELmD	Enchilada
🟥 FASER	🟥 HKC	HS1-SIV	ICEPOLE	🟩 Joltik
Julius	🟩 KIASU	Ketje	Keyak	LAC
MORUS	🟦 Marble	🟥 McMambo	Minalpher	NORX
OCB	OMD	PAEQ	🟥 PAES	🟥 PANDA
🟥 POET[1]	POLAWIS	PRIMATEs	🟩 Prøst	Raviyoyla
SCREAM	🟦 SHELL	SILC	STRIBOB	Sablier
Silver	Tiaoxin	TriviA-ck	Wheesht	YAES
iFeed[AES]	π-Cipher			

[1] Only POET-G withdrawn.
🟥 Withdrawn, 🟩 COPA-Variants, 🟦 Indirect COPA-Variants.

4.3.3. Classification of CAESAR Candidates

Figure 4.2 gives a classification of the first round candidates. Thereby, we omit the schemes that are already withdrawn from the competition. The data for the statistics is based on the classification summary of Abed *et al.* [AFL15b], the AE zoo [AKL$^+$] and a visualization of Andreeva [AD]. For a detailed classification of first round candidates we refer to [AFL15b].

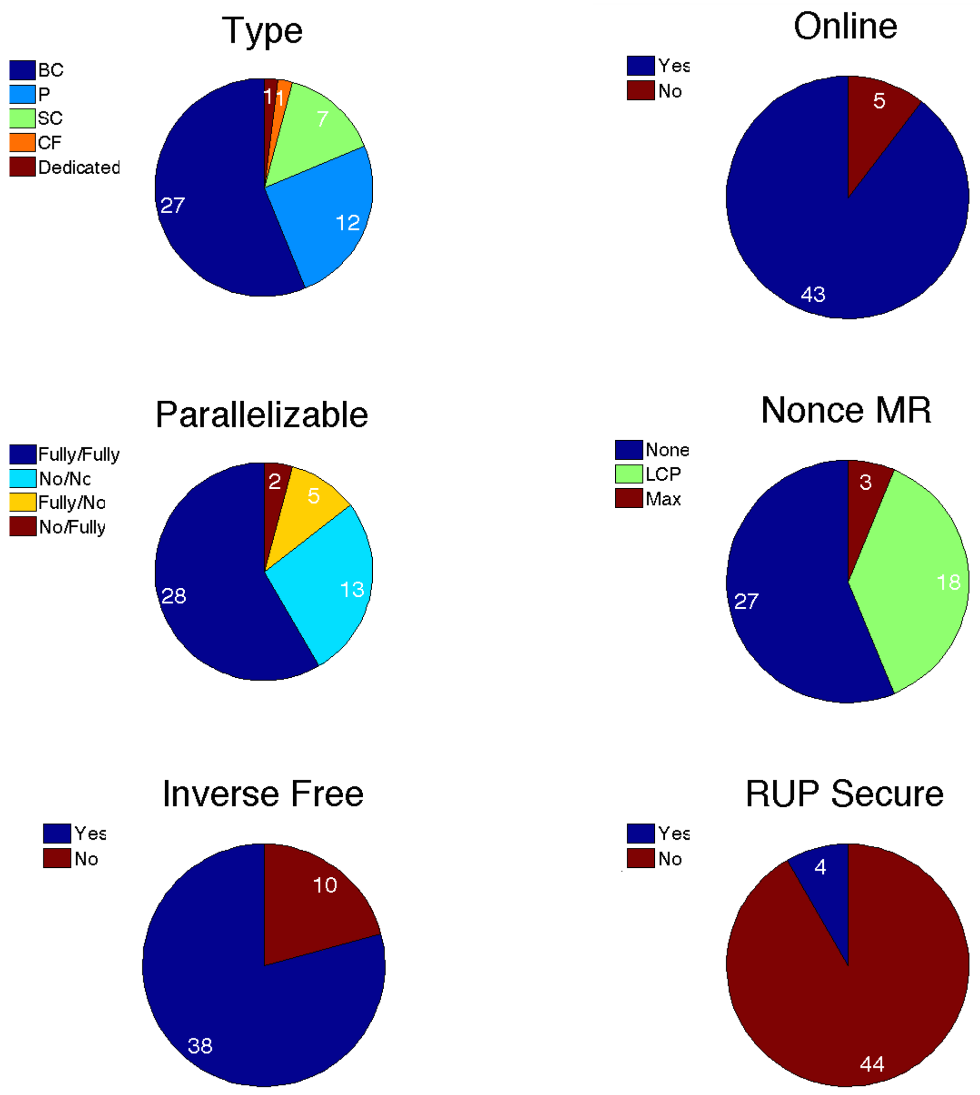


Figure 4.2.: Classification of First Round Candidates. (**tl.**) Different building blocks of AE, where BC denotes a block cipher, P a permutation, SC a stream cipher, CF a compression function and Dedicated a dedicated function, respectively. (**tr.**) Online or Offline AE. (**ml.**) Parallelizability denoted for $\mathcal{E}/\mathcal{D}$. (**mr.**) Resistance against nonce-reuse – completely denoted by Max, none as None or up to the longest common prefix as LCP. (**bl.**) Existence of an inverse of $\mathcal{E}$. (**br.**) Secure for RUP.

5 Provable Security

In this chapter, we introduce provable security. In detail, we define the approach of reductionist security to generate a security proof under reasonable assumptions. Moreover, we illustrate some well known notions for security mainly based on the topic of authenticated encryption in the symmetric key setting. Finally, we discuss different proof techniques and introduce Patarin's coefficient-H technique – used in our proofs for *COPA* in Chapter 6.

5.1. Preliminaries

A cryptographic scheme should not resist only one particular attack, but rather be resistant to a class of attacks (*e.g.* attacks that use limited resources). In this setting, we cover not only currently known attacks, but also yet unknown attacks – which may be discovered in the future. To overcome this problem, one can use the approach of *reductionist security*. Reductionist security was established in the early 1980s and is based on complexity theory. Thereby, a cryptographic scheme is called *provable secure* if there is a formal description of its security requirements in an adversarial model with clear assumptions on the computational resources of the adversary. Then the proof of security states that an adversary, must satisfy those security requirements (*i.e.* solve a certain problem) in order to break the security of the cryptographic scheme. A security proof is then a polynomial reduction to a well known problem.

Classical Approach to Provide Security to a Cryptographic Scheme

In the traditional setting, security of a cryptographic scheme was determined to be secure against state-of-the-art cryptanalytic attacks. The approach to design a cryprographic scheme was as follows:

1. Design/(Re-design) the scheme.
2. Perform cryptanalysis.
3. If flaw (*i.e.* break of scheme) – go to 1.

Provable Security Approach

The *provable security* approach (also called *reductionist security*) clearly defines a class of adversaries $\mathcal{C}$ to break a abstract scheme $\mathcal{S}$ under a certain attack model $\mathcal{M}$. The approach works in four steps:

1. Select a class of adversaries (*e.g.* computationally bound or with unlimited resources).
2. Develop an abstract model of the cryptographic scheme (*e.g.* authenticated encryption scheme).
3. Specify a security definition (*e.g.* IND-CCA, INT-CTXT).
4. Prove secure under reasonable assumptions, by the use of complexity theory (*e.g.* a block cipher is a secure PRP).

More formally, select an adversary $\mathcal{A} \in \mathcal{C}$, derive a model of the scheme $\mathcal{S}$ and choose an attack model $\mathcal{M}$. The proof must show that $\mathcal{A}$ can only break the security of $\mathcal{S}$ in the model $\mathcal{M}$ if it is able to solve a certain problem $\mathcal{P}$, otherwise it is secure. Therefore, the problem $\mathcal{P}$ is often polynomially reduced to a well known problem $\mathcal{P}_1$. $\mathcal{P}$ can be reduced to $\mathcal{P}_1$ if we can define a mapping between those two problems such that if we solve problem $\mathcal{P}_1$, we also solve $\mathcal{P}$. We denote a reduction as $\mathcal{P} \leq_P \mathcal{P}_1$ and say problem $\mathcal{P}$ is *polynomial reducible* to problem $\mathcal{P}_1$. Now *hardness* of $\mathcal{P}$ implies *hardness* of $\mathcal{P}_1$ (*i.e.* $\mathcal{P} \Rightarrow \mathcal{P}_1$).

In order to prove a scheme $\mathcal{S}$ secure we derive a cryptographic assumption P and reduce it to the security of the scheme S. As long as P holds S is secure (*i.e.* $P \Rightarrow S$). This is equal to

$$\begin{aligned}\forall \mathcal{A}\ \exists \mathcal{B} : \mathbf{Adv}_S(\mathcal{A}) &\leq \mathbf{Adv}_P(\mathcal{B})\\ \textbf{or } Pr[\mathcal{A} \text{ breaks } S] &\leq Pr[\mathcal{B} \text{ breaks } P]\end{aligned}$$

Application to Asymmetric Cryptography. Provable security was first developed in the asymmetric cryptography setting. It is based on hard mathematical problems like the hardness of the *integer factorization problem* or the *discrete logarithm problem*.

Application to Symmetric Cryptography. Later on, symmetric cryptography made also use of the *provable security* approach. Provable security is in this setting based on the reduction to well known primitives (*e.g.* `AES`, `Keccak`) in the standard model or to some idealized primitives (*e.g.* PRF, PRP, One-way functions) in the ideal world model. In this thesis, we focus only on *provable security* for symmetric cryptography.

5.1.1. Standard Model

Proofs are accomplished in an underlying model of computation. In the standard model, the adversary $\mathcal{A}$ is only bound by the amount of time and computational power that is available. Then, a cryptographic scheme that can be proven secure in the standard model, if it uses only standard complexity theoretic assumptions on the primitive. In general, security proofs in the standard model are hard to accomplish. Therefore, a lot of proofs assume idealized primitives. Nevertheless, proofs accomplished in the standard model are stronger than proofs in ideal world model.

5.1.2. Ideal World Model

The ideal world model, also called *random oracle model* was first introduced by Bellare and Rogaway [BR93]. In this computational model, cryptographic primitives are replaced by some idealized version of the primitive. Examples for such idealized primitives are a *one-way* function or a *random oracle*. The latter is just a theoretical black box, that returns for every different input a string chosen uniformly at random from the output range. Many proofs operate in this kind of computational model as otherwise it is hard to achieve a security proof without the assumption of an idealized primitive – as it is done in the standard model.

5.2. Notions of Security

In the following, we introduce some well known notions of security. The definition of security (*e.g.* privacy, integrity or authenticity) of a cryptographic scheme is given by a notion. A notion thereby represents what an adversary has to achieve to break the scheme. Such security notions need to be defined rigorously such that they can be used for a wide range of cryptographic schemes.

First, we start with some general definitions and assumptions on security. We fix an adversary $\mathcal{A}$, which represents a PPT algorithm. $\mathcal{A}$ is computationally bounded in time, number of queries and length of these queries. Moreover, we fix a distinguisher $\mathcal{D}$, that has the same properties as $\mathcal{A}$, but tries to distinguish between two worlds. $\mathcal{O}$ denotes any oracle, \$ denotes the random oracle. We write $\mathcal{D}^{\mathcal{O}}$ and mean that $\mathcal{D}$ accesses oracle $\mathcal{O}$. For the following notions we refine the distinguishing attack.

Distinguishing Attack. $\mathcal{D}$ has access to an oracle $\mathcal{O}$ that can either be the cryptographic scheme or an idealized version (*e.g.* the random oracle \$). In the beginning, the environment, which $\mathcal{D}$ interacts with, tosses a fair coin to select a bit b. Following that, $\mathcal{D}$ submits its queries to either the cryptographic scheme or the random oracle, depending on the outcome of the coin toss for bit b. $\mathcal{D}$ succeeds if it can determine the value of b.

The distinguishing-advantage is given by

$$\mathbf{Adv}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{O}_1} \Rightarrow 1] - Pr[\mathcal{D}^{\mathcal{O}_2} \Rightarrow 1]\right|$$

where the oracles $\mathcal{O}_1, \mathcal{O}_2$ represents the two worlds, that $\mathcal{D}$ wants to distinguish. The advantage $\mathbf{Adv}(q, \ell, t)$ is computational bound by the number of queries q each with length ℓ, and a total length of σ for all queries and runs in time t. Furthermore, we assume that $\mathcal{D}$ does not submit any trivial query, from which it already knows the answer (*i.e.* a query to a decryption oracle $\mathcal{E}^{-1}$ with results from a previous query to an encryption oracle $\mathcal{E}$).

For the following security notions, we fix $\mathcal{AE} = (\mathcal{K}, \mathcal{E}, \mathcal{E}^{-1})$ an authenticated encryption scheme, where $\mathcal{E}$ denotes the encryption oracle and $\mathcal{E}^{-1}$ denotes the decryption oracle. When we use a nonce-respecting adversary we assume that it never submits two queries with the same nonce N.

5.2.1. Privacy

For *privacy* we know four notions of security. While for the IND-CPA notion $\mathcal{D}$ is not allowed to reuse nonces at any point, in the IND-CCA notions $\mathcal{D}$ is allowed to reuse nonces for decryption queries.

IND-CPA. Is the notion for *indistinguishability* under a chosen-plaintext attack. $\mathcal{D}$ has access to either the *real* encryption oracle $\mathcal{E}$ or a random oracle \$. Its advantage is given by

$$\mathbf{Adv}^{\text{IND-CPA}}_{\mathcal{AE}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$(\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}^{\text{IND-CPA}}_{\mathcal{AE}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all IND-CPA adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries.

IND-CCA1. Is the notion for *indistinguishability* under a non-adaptive chosen-ciphertext attack. $\mathcal{D}$ has access to either the *real* encryption oracle

$\mathcal{E}$ and decryption oracle $\mathcal{E}^{-1}$ or a random oracle \$ and the decryption oracle $\mathcal{E}^{-1}$. Its advantage is given by

$$\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA1}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot),\mathcal{E}_K^{-1}(\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$(\cdot),\mathcal{E}_K^{-1}(\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA1}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all IND-CPA adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries.

IND-CCA2. Is the notion for *indistinguishability* under an adaptive chosen-ciphertext attack. $\mathcal{D}$ has access to either the *real* encryption oracle $\mathcal{E}$ and decryption oracle $\mathcal{E}^{-1}$ or a random oracle \$ and the decryption oracle $\mathcal{E}^{-1}$. Its advantage is given by

$$\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA2}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot),\mathcal{E}_K^{-1}(\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$(\cdot),\mathcal{E}_K^{-1}(\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA2}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all IND-CPA adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries.

IND-CCA3. Is another notion for *indistinguishability* under an adaptive chosen-ciphertext attack, whereas Shrimpton [Shr04] proves that an encryption scheme achieving this kind of notion is an authenticated encryption scheme. It is similar to *IND-CCA2* but with the difference that $\mathcal{D}$ in the bogus world has only access to a decryption oracle that *always* returns $\perp$. Then $\mathcal{D}$ has access to either the *real* encryption oracle $\mathcal{E}$ and decryption oracle $\mathcal{E}^{-1}$ or a random oracle \$ and a bogus decryption oracle $\perp(\cdot)$ that always returns INVALID (*i.e.* $\perp$). Its advantage is given by

$$\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA3}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot),\mathcal{E}_K^{-1}(\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$(\cdot),\perp(\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}_{\mathcal{AE}}^{\text{IND-CCA3}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all IND-CPA adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries.

5.2.2. Integrity

There are two notions of security with regards to *integrity*.

INT-PTXT. Is a notion for *integrity* of plaintexts. $\mathcal{A}$ plays the game INT-PTXT$^{\mathcal{A}}_{\mathcal{AE}}$ (see Algorithm 5.1) and wins if its outputs *True*. Its advantage is given by

$$\mathbf{Adv}^{\text{INT-PTXT}}_{\mathcal{AE}}(\mathcal{A}) = Pr[\text{INT-PTXT}^{\mathcal{A}}_{\mathcal{AE}} \Rightarrow True]$$

where by $\mathbf{Adv}^{\text{INT-PTXT}}_{\mathcal{AE}}(q,\ell,\sigma,t)$, we denote the maximal advantage taken over all INT-PTXT adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries. $\mathcal{A}$ wins game INT-PTXT$^{\mathcal{A}}_{\mathcal{AE}}$ if it successfully submits a ciphertext C that decrypts to a valid message $M \neq \perp$, where M was never submitted to $\mathcal{E}_K(\cdot)$. Therefore, the notion states that it is computationally infeasible for any $\mathcal{A}$ to generate such a C.

INT-CTXT. Is a notion for *integrity* of ciphertexts. $\mathcal{A}$ plays the game INT-CTXT$^{\mathcal{A}}_{\mathcal{AE}}$ (see Algorithm 5.1) and wins if its outputs *True*. Its advantage is given by

$$\mathbf{Adv}^{\text{INT-CTXT}}_{\mathcal{AE}}(\mathcal{A}) = Pr[\text{INT-CTXT}^{\mathcal{A}}_{\mathcal{AE}} \Rightarrow True]$$

where by $\mathbf{Adv}^{\text{INT-CTXT}}_{\mathcal{AE}}(q,\ell,\sigma,t)$, we denote the maximal advantage taken over all INT-CTXT adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries. $\mathcal{A}$ wins game INT-CTXT$^{\mathcal{A}}_{\mathcal{AE}}$ if it successfully submits a ciphertext C that decrypts to a valid message $M \neq \perp$, where C was never outputted from $\mathcal{E}_K(\cdot)$. Therefore, the notion states that it is computationally infeasible for any $\mathcal{A}$ to generate such a C. Since there is no limitation on the submitted plaintext M, this notion is stronger than INT-PTXT.

Algorithm 5.1: Game INT-PTXT$^{\mathcal{A}}_{\mathcal{AE}}$ (left) and INT-CTXT$^{\mathcal{A}}_{\mathcal{AE}}$ (right).

```
Game INT-PTXT$^{\mathcal{A}}_{\mathcal{AE}}$
  procedure Initialize
    $K \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$
  procedure Encrypt(M)
    $C \xleftarrow{\$} \mathcal{E}_K(M)$; $S \leftarrow S \cup \{M\}$;
    return $C$
  procedure Decrypt(C)
    $M \leftarrow \mathcal{E}_K^{-1}(C)$
    if $(M \neq \perp$ and $M \notin S)$ then
      win $\leftarrow$ True
    return win
  procedure Finalize
    return win
```

```
Game INT-CTXT$^{\mathcal{A}}_{\mathcal{AE}}$
  procedure Initialize
    $K \xleftarrow{\$} \mathcal{K}$; $S \leftarrow \emptyset$
  procedure Encrypt(M)
    $C \xleftarrow{\$} \mathcal{E}_K(M)$; $S \leftarrow S \cup \{C\}$;
    return $C$
  procedure Decrypt(C)
    $M \leftarrow \mathcal{E}_K^{-1}(C)$
    if $(M \neq \perp$ and $C \notin S)$ then
      win $\leftarrow$ True
    return win
  procedure Finalize
    return win
```

5.2.3. Additional Security Notions

In this section, we describe some additional security notions used for the proofs in Chapter 6. We first define the notion of MRAE, followed by the OAE1 notion. Online authenticated encryption reduces memory usage and latency for applications by processing the input as it arrives to the encryption/decryption device. For the proof of $\overline{COPA}$ in Section 6.3.2 we use the OAE1 notion.

MRAE. The notion of misuse-resistant authenticated encryption was introduced by Rogaway and Shrimpton in [RS06]. They propose a notion to achieve AE-security for an ivE-scheme, even if the IV gets reused. Therefore, let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{E}^{-1})$ denote an IV-based $\mathcal{SE}$ scheme. Then, the MRAE-advantage of $\mathcal{D}$ is denoted as

$$\mathbf{Adv}_{\Pi}^{\mathrm{MRAE}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot,\cdot,\cdot),\mathcal{E}_K^{-1}(\cdot,\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$(\cdot,\cdot,\cdot),\perp(\cdot,\cdot,\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}_{\Pi}^{\mathrm{MRAE}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all MRAE adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries. Furthermore, we denote the calls to the encryption oracle as $\mathcal{E}_K(IV, AD, M)$ and to the decryption oracle as $\mathcal{E}_K^{-1}(IV, AD, C)$. $\mathcal{D}$ is not allowed to query $\mathcal{E}_K^{-1}$ with the results of a previous query to $\mathcal{E}_K$.

OAE1. To provide a notion between strict nonce-respecting nAE and nonce-reusing MRAE, Fleischman *et al.* introduced OAE1 in [FFL12]. Consider $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{E}^{-1})$ as an OAE1-scheme [AFL+15a], where $\$^{\mathrm{OAE}}$ returns a random online string of length $|\mathcal{E}_K(M)|$. Then, the OAE1-advantage is given by

$$\mathbf{Adv}_{\Pi}^{\mathrm{OAE1}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K(\cdot,\cdot,\cdot),\mathcal{E}_K^{-1}(\cdot,\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\$^{\mathrm{OAE}}(\cdot,\cdot,\cdot),\perp(\cdot,\cdot,\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from the key $K \xleftarrow{\$} \mathcal{K}$, and random coins of the random oracle \$ or $\mathcal{D}$, if any. By $\mathbf{Adv}_{\Pi}^{\mathrm{OAE1}}(q, \ell, \sigma, t)$, we denote the maximal advantage taken over all OAE1 adversaries that run in time t and make at most q queries with ℓ the length in blocks for each query and σ the total length of all queries. Encryption calls are denoted as $\mathcal{E}_K(N, AD, M)$ and decryption calls $\mathcal{E}_K^{-1}(N, AD, C)$. Additionally, we define the advantage of $\mathcal{D}$ distinguishing Π from an online permutation *OPerm*. Then, the OPERM-CCA advantage is given by

$$\mathbf{Adv}_{\Pi}^{\mathrm{OPERM\text{-}CCA}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\mathcal{E}_K,\mathcal{E}_K^{-1}} \Rightarrow 1] - Pr[\mathcal{D}^{\pi,\pi^{-1}} \Rightarrow 1]\right|$$

where the probabilities are taken from $K \xleftarrow{\$} \mathcal{K}$ and $\pi \xleftarrow{\$} OPerm_n$, where $OPerm_n$ denotes the set of all length-preserving permutations π on $(\{0,1\}^n)^*$ and the i^{th} output block of $\pi(\cdot)$ depends only on the input from 1 to i.

In the following, we define the notions for SPRP and $\widetilde{SPRP}$ used in the proofs in Section 6.3.2 and Section 6.6.2, respectively.

Definition 18. *(Strong Pseudo Random Permutation): A strong pseudo-random permutation on n-bits is a efficient, keyed map from $p : \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$, such that a PPT distinguisher can distinguish between p, p^{-1} and an ideal* *random permutation π, π^{-1}, with only a negligible probability. $\pi \xleftarrow{\$} Perm(n)$, where $Perm(n)$ denotes the set of all permutations on n-bits.*

SPRP. We fix a $\mathcal{SE}$-scheme $E_K(\cdot), E_K^{-1}(\cdot)$ and let $\pi(\cdot), \pi^{-1}(\cdot)$ denote an ideal random permutation. Then, we define the SPRP-advantage as

$$\mathbf{Adv}_E^{\text{SPRP}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{E_K(\cdot),E_K^{-1}(\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\pi(\cdot),\pi^{-1}(\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from $K \xleftarrow{\$} \mathcal{K}$ and $\pi \xleftarrow{\$} Perm(n)$. Moreover, we define $\mathbf{Adv}_E^{\text{SPRP}}(q,t)$ as the maximum advantage over all *SPRP* adversaries $\mathcal{A}$ on E that run in time t and make at most q queries.

Definition 19. *(Tweakable Strong Pseudo Random Permutation): A tweakable strong pseudo-random permutation on n-bits is a efficient, keyed map from $p : \{0,1\}^k \times \{0,1\}^t \times \{0,1\}^n \to \{0,1\}^n$, such that a PPT distinguisher can distinguish between p, p^{-1} and an ideal tweakable random permutation $\widetilde{\pi}, \widetilde{\pi}^{-1}$, with only a negligible probability. $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T}, n)$, where $Perm(\mathcal{T}, n)$ denotes the set of all mappings from $\mathcal{T}$ to permutations on n-bits.*

$\widetilde{SPRP}$**.** We fix a tweakable $\mathcal{SE}$-scheme $\widetilde{E_K}(\cdot,\cdot), \widetilde{E_K}^{-1}(\cdot,\cdot)$ and let $\widetilde{\pi}(\cdot), \widetilde{\pi}^{-1}(\cdot)$ denote an ideal tweakable random permutation. Then, we define the $\widetilde{SPRP}$-advantage as

$$\mathbf{Adv}_{\widetilde{E}}^{\widetilde{SPRP}}(\mathcal{D}) = \left|Pr[\mathcal{D}^{\widetilde{E_K}(\cdot,\cdot),\widetilde{E_K}^{-1}(\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{D}^{\widetilde{\pi}(\cdot,\cdot),\widetilde{\pi}^{-1}(\cdot,\cdot)} \Rightarrow 1]\right|$$

where the probabilities are taken from $K \xleftarrow{\$} \mathcal{K}$ and $\widetilde{\pi} \xleftarrow{\$} Perm(\mathcal{T}, n)$. Moreover, we define $\mathbf{Adv}_{\widetilde{E}}^{\widetilde{SPRP}}(q,t)$ as the maximum advantage over all $\widetilde{SPRP}$ adversaries $\mathcal{A}$ on $\widetilde{E}$ that run in time t and make at most q queries.

5.2.4. Others

Moreover, there are several other notions for the security of a cryptographic schemes. For *indistinguishability* we often use IND-PRF to distinguish a MAC from a PRF and IND-PRP to distinguish a block cipher from a PRP. Moreover, there are notions for *unforgeability* of a MAC and *non-malleability* of a $\mathcal{SE}$-scheme.

5.3. Relations between Notions

In Figure 5.1 we denote the relation between the notions for a symmetric encryption scheme $\mathcal{SE}$. More precisely, we show how the notions for privacy (*i.e.* indistinguishability and non-malleability) and integrity (*i.e.* integrity of plaintext/ciphertexts and weak/strong forgery) interact together. Therefore, we denote $A \rightarrow B$ as *implication* between notions A and B and $A \not\rightarrow B$ as *separation*, respectively. For more details about the relations we refer to [BN00].

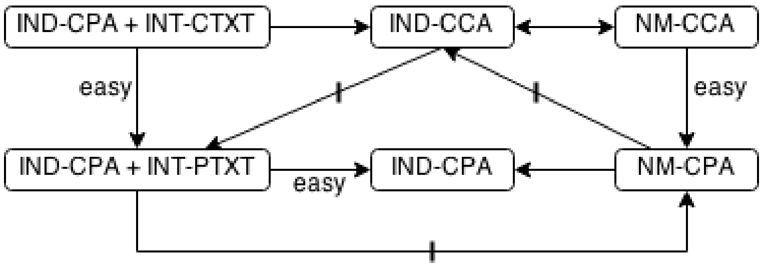


Figure 5.1.: Relations between Notions for Symmetric Encryption.

It is easy to see that IND-CPA + INT-CTXT implies IND-CPA + INT-PTXT since INT-CTXT is a stronger notion than INT-PTXT. Moreover, IND-CPA + INT-CTXT implies IND-CPA and NM-CCA implies NM-CPA.

IND-CPA + INT-CTXT $\Rightarrow$ IND-CCA2. In the following theorem, we show that a symmetric encryption scheme that is IND-CPA secure and achieves the notion of INT-CTXT also achieves IND-CCA2.

Theorem 1. *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ denote a symmetric encryption scheme and $\mathcal{A}$ denote an IND-CCA2 adversary, then we can construct two adversaries – $\mathcal{A}_{ctxt}$ and $\mathcal{A}_{cpa}$ such that,*

$$\mathbf{Adv}_{\mathcal{SE}}^{\text{IND-CCA2}}(\mathcal{A}) \leq 2 \cdot \mathbf{Adv}_{\mathcal{SE}}^{\text{INT-CTXT}}(\mathcal{A}_{ctxt}) + \mathbf{Adv}_{\mathcal{SE}}^{\text{IND-CPA}}(\mathcal{A}_{cpa})$$

where $\mathcal{A}$ and $\mathcal{A}_{ctxt}$ runs in time t and makes q_e queries to its encryption oracle and q_d queries to its decryption oracle and $\mathcal{A}_{cpa}$ makes q_e queries to its encryption oracle.

We omit the proof in this thesis and refer to [BN00].

IND-CCA3 $\Leftrightarrow$ AE. Shrimpton [Shr04] proved that IND-CCA3 implies AE and vice versa.

Theorem 2. *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ denote a symmetric encryption scheme and $\mathcal{A}$ denote an IND-CCA3 adversary, then we can construct two adversaries – $\mathcal{A}_{ctxt}$ and $\mathcal{A}_{cpa}$ such that,*

$$\boldsymbol{Adv}^{\textit{IND-CPA}}_{\mathcal{SE}}(\mathcal{A}_{cpa}) \leq \boldsymbol{Adv}^{\textit{IND-CCA3}}_{\mathcal{SE}}(\mathcal{A})$$

$$\boldsymbol{Adv}^{\textit{INT-CTXT}}_{\mathcal{SE}}(\mathcal{A}_{ctxt}) \leq 2 \cdot \boldsymbol{Adv}^{\textit{IND-CCA3}}_{\mathcal{SE}}(\mathcal{A})$$

where $\mathcal{A}$, $\mathcal{A}_{cpa}$ and $\mathcal{A}_{ctxt}$ run in time t and make q queries of total length μ for the first statement and $\mathcal{A}$ runs in time $t' = t + O(\mu)$, makes $q' = q + 1$ queries of total length $\mu' = \mu + 1$ for the second statement.

We omit the proof in this thesis and refer to [Shr04].

Theorem 3. *Let $\mathcal{SE} = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ denote a symmetric encryption scheme and $\mathcal{A}$ denote an IND-CCA3 adversary, then we can construct two adversaries – $\mathcal{A}_{ctxt}$ and $\mathcal{A}_{cpa}$ such that,*

$$\boldsymbol{Adv}^{\textit{IND-CCA3}}_{\mathcal{SE}}(\mathcal{A}) \leq \boldsymbol{Adv}^{\textit{IND-CPA}}_{\mathcal{SE}}(\mathcal{A}_{cpa}) + q \cdot \boldsymbol{Adv}^{\textit{INT-CTXT}}_{\mathcal{SE}}(\mathcal{A}_{ctxt})$$

where $\mathcal{A}$, $\mathcal{A}_{cpa}$ run in time t and $\mathcal{A}_{ctxt}$ runs in time $t' = t + O(q)$ and all make q queries of total length μ.

We omit the proof in this thesis and refer to [Shr04].

5.4. Proving Techniques

In order to get reliable, robust and especially secure cryptographic schemes a designer has to provide a security proof under reasonable assumptions. There are several different techniques to obtain a proof of security. Most popular are the techniques such as *polynomial reduction* or *game-based proofs*. However, there are also other techniques like a *hybrid argument* or *Patarin's coefficient-H technique*.

5.4.1. Polynomial Reduction

As already mentioned in Section 5.1, in the case of *polynomial reduction* the designer of a cryptographic scheme $\mathcal{S}$ reduces the security of $\mathcal{S}$ to a well known problem $\mathcal{P}$, where $\mathcal{P}$ is normally the representation of an idealized cryptographic primitive (*e.g.* PRF, PRP, One-Way Function).

5.4.2. Game-based proofs

Another technique to obtain a security proof is to use game theory. There, a security proof is based on game-playing. Bellare and Rogaway use this technique quite often it their security proofs. In the following, we give a short introduction to the game-based proof technique. For a detailed description we refer to [BR04].

We denote a game G played by adversary $\mathcal{A}$ as $G^{\mathcal{A}}$, and the output of G as $G^{\mathcal{A}} \Rightarrow out$. We assume that boolean flags are initialized to *False*, sets are initialized to the empty set $\emptyset$ and games G_i, G_j are *identical until* bad, iff the code of both games G_i, G_j only differs in statements after flag bad is set to *True*.

Procedures. A game G consists of several procedures. In this thesis, procedures are denoted in capital writing style. Therefore, a game consists at least of an INITIALIZE and a FINALIZE procedure. Moreover, it can contain additional procedures to handle queries of an adversary $\mathcal{A}$. Now, G starts by calling the procedure INITIALIZE and the outputs of these operations are input to $\mathcal{A}$. Then, $\mathcal{A}$ executes and its oracle queries are handled via the corresponding procedures in G. If $\mathcal{A}$ terminates, the output values of $\mathcal{A}$ are input to the FINALIZE procedure.

Fundamental Lemma of Game-Playing. The fundamental lemma of game-playing defines the basis behind every game-playing proof.

Lemma 1. *Let G_i and G_j denote two games, that are identical until* bad*. Moreover, let $\mathcal{A}$ denote an adversary. Then,*

$$Pr[G_i^{\mathcal{A}} \Rightarrow out] - Pr[G_j^{\mathcal{A}} \Rightarrow out] \leq Pr[G_j^{\mathcal{A}} \text{ sets } \mathsf{bad}].$$

We omit the proof in this thesis and refer to [BR04].

A proof of security is established in the following way. First, we define one or more games, related with the security property to be proven. A game can be seen as a challenge involving the adversary and oracles provided by the environment. Then, an adversary is successful if she wins the game. Additionally, using the fundamental lemma of game-playing, we can relate games that are identically until bad.

5.4.3. Hybrid Argument

Besides the already mentioned paradigms, another technique is the usage of the triangle inequality. In order to solve a problem $\mathcal{P}$ it is not always convenient to reduce it straightforward to a problem P_1 but sometimes easier and better understandable to fellow cryptographers (*e.g.* verifier) to reduce it over a series of other problems $\mathcal{P}_i$ to the final problem $\mathcal{P}_n$. Then, we use the triangle inequality to show that we achieve $\mathcal{P} \leq_P \mathcal{P}_n$ by showing that $\mathcal{P} \leq_P \sum_i P_i \leq_P \mathcal{P}_n$.

5.4.4. Patarin's Coefficient-H Technique

The coefficient-H technique was introduced by Patarin in [Pat08] and refined by Chen and Steinberger in [CS14]. It works as follows. Consider a deterministic distinguisher $\mathcal{D}$ that has access to one or more oracles $\mathcal{O}$. For each of theses oracles we assign a *world* – the ideal world and the real world. We denote the oracle for the ideal world as \$ and for the real world as $\mathcal{E}$. Moreover, we define a view ν as a transcript of a set of queries from the interaction with oracle $\mathcal{E}$ or \$. In other words, a view ν is the set of query-response tuples (*i.e.* $\nu = (M_i, C_i)_{i=1}^{q}$, with $1 \leq i \leq q$).

We partition views into **good** and **bad** views, where $\nu \in \nu_{good} \cup \nu_{bad}$. Furthermore, for each oracle $\mathcal{O}$ we define a probability distribution X such that a view $\nu \in X$, when the interaction of a query q_i with oracle $\mathcal{O}$ yields this view ν and the probability $Pr(\nu \in X) > 0$.

Then, the distinguishing advantage of $\mathcal{D}$ is defined as

$$\Delta(X,Y) = \left|Pr[\mathcal{D}^{\mathcal{E}} \Rightarrow 1] - Pr[\mathcal{D}^{\$} \Rightarrow 1]\right|$$

where X, Y are the probability distributions of $\mathcal{E}$ and \$ and $\Delta(X,Y)$ denotes the statistical distance between those two sets. The statistical distance is lower bounded by $\delta + \epsilon$, with $0 \leq \epsilon \leq 1$ if for all views $\nu \in \nu_{good}$

$$\frac{Pr(\nu(\mathcal{D}^{\mathcal{E}}) = \nu)}{Pr(\nu(\mathcal{D}^{\$}) = \nu)} \geq 1 - \epsilon \tag{5.1}$$

and for views $\nu \in \nu_{bad}$

$$Pr(\nu(\mathcal{D}^{\$}) \in \nu) \leq \delta$$

We omit the proof for these statements in this thesis and refer to [Pat08] and [CS14]. The *idea* behind this technique is that there are only a few views that are more likely to appear in \$ than in $\mathcal{E}$ – bringing the ratio (5.1) close to one. These views are the *bad* views that are lower bounded by δ.

5.4.5. Tightness

First we give the definition of *tightness* and continue with some remarks on how to achieve a tight reduction and what it means if we have a non-tight reduction.

Consider a cryptographic scheme $\mathcal{S}$. Related with the security proof is the problem $\mathcal{P}$ that an adversary $\mathcal{A}$ has to solve in order to break the cryptographic scheme. Furthermore, suppose $\mathcal{A}$ has access to oracle $\mathcal{O}$ that fulfills the adversarial goal – specified by the security definitions of $\mathcal{S}$. We denote T as the time, and ϵ as the probability that $\mathcal{A}$ takes in order to get a successful outcome. Additionally, we denote T' as the time $\mathcal{A}$ needs to solve $\mathcal{P}$ with success probability ϵ'. Then, a security reduction is called *tight* if $T' \approx T$ and $\epsilon' \approx \epsilon$. More or less it is called *non-tight* if $T' \gg T$ or $\epsilon' \gg \epsilon$.

A *tight* proof is desirable – as then we can assume that breaking scheme $\mathcal{S}$ is as hard as solving problem $\mathcal{P}$. Nevertheless, it is not always possible to achieve a tight reduction, whereas it is possible to apply a hybrid argument to obtain a bound. The more loose the boundary is, the *tightness gap* grows and also the **Adv** of any adversary grows.

6 Provable Security of Submissions to the CAESAR Competition

In this chapter, we state our results of this thesis. We have selected the AE composition scheme *COPA* [ABL+13], where we prove a variant of this scheme $\overline{COPA}$ secure beyond the birthday bound. Moreover, we studied the application of $\overline{COPA}$ to CAESAR candidates, that rely heavily on the *COPA* design (*i.e.* AES-*COPA* [ABL+14b], PRØST-*COPA* [KLL+14], Deoxys [JNP14b], Joltik [JNP14c] and KIASU [JNP14d]). For the former two candidates our proof in Section 6.3.2 can be applied without any restrictions. However, for the latter three candidates we have developed an additional proof given in Section 6.6.2 and 6.6.3.

6.1. Preliminaries

Modern cryptography is build upon provable secure cryptographic schemes. 33 out of the 48 remaining candidates of the CAESAR competition are

supported with a security proof [AFL15b]. To justify the use of a CAESAR candidate and to point out potential weaknesses these candidates need third-party cryptanalysis and security proofs to show robustness in presence of a potential adversary. Therefore, we use the framework of provable security to provide such a proof to a variant of the AE scheme *COPA* – called $\overline{COPA}$.
Our motivation for the choice of *COPA* was, that it features the first fully parallelizable online authenticated encryption scheme – for encryption and decryption. Furthermore, it offers nonce-misuse resistance, and therefore providing security against IND-CPA attacks up to the birthday bound, even if nonces get reused. The usage of doubling in the tweak values provides *COPA* with the `XE` and `XEX` constructions of Rogaway [Rog04a] – for simple and easy to analyze proofs. Additionally, it offers an exceptional software performance and is up to 5 times faster than McOE-G [FFL12], TC1 or TC3 [RZ11]. Moreover, *COPA* is used in several CAESAR submissions (*e.g.* AES-*COPA*, PRØST-*COPA*, Deoxys, Joltik and KIASU).

6.1.1. Recent Attacks on OAE Schemes

In the following, we consider only online authenticated encryption schemes. Most recently, there were some new attacks against OAE introduced. All these attacks operate in the nonce-reusing setting. Hoang *et al.* [HRRV15] proposed a *trivial attack*, by reducing the block size of the underlying block cipher to a small number (*e.g.* $n = 1$). Reyhanitabar *et al.* proposed a more involved attack called *Chosen-Prefix Secret-Suffix Attack (CPSS)* [RV14] and Abed *et al.* introduced a similar attack called *Chosen-Position Overwrite-Secret Attack (CPOS)* [AFL+15a]. Since our candidate scheme $\overline{COPA}$ operates in the nonce-respecting setting, these attacks doesn't apply to our scheme.

6.2. COPA

In this section, we give a short description of *COPA* and its security bounds. *COPA* is an AE composition mode introduced by Andreeva *et al.* [ABL+13] at Asiacrypt 2013 and builds upon the parallelizable online cipher *COPE* to provide privacy and authenticity – authenticated encryption.

6.2.1. Description of COPA

COPA takes as input an arbitrary but finite-length associated data AD and message M, that are split into n-bit blocks (*i.e.* $A = (A_1 \dots A_d)$ and $M = (M_1 \dots M_\ell)$, where each $|A_i| = |M_i| = n$). For messages that are not a multiple of the block length *tag splitting* is applied in case of $|M| < n$, and *XLS* for the case of $|M| > n$. For associated data that is not a multiple of the block length, the last A block is padded using the $10^\star$ *padding*. It outputs a ciphertext C, split into $\ell \times n$-bit blocks $C_1 \dots C_\ell$ and a tag T of length $|T| = n$. The message and associated data processing is given in Algorithm A.1 and Algorithm A.2 in Appendix A. Moreover, the encryption process is illustrated in Figure 6.1 and 6.2.

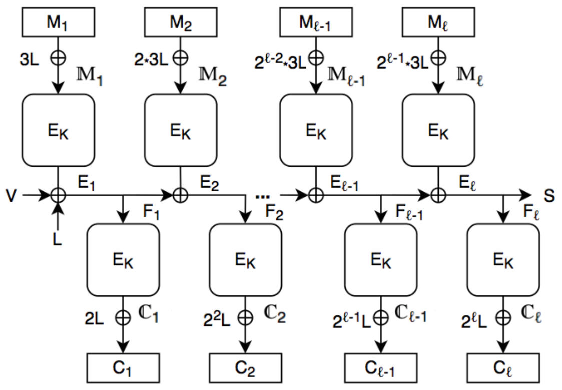


Figure 6.1.: Message Processing for COPA.

Notation of COPA. In the following, we give the notation of *COPA* used throughout this thesis. We denote every value as an instance of a query (*e.g.* nonce N is denoted as N_i for query q_i). Index i runs between $1 \leq i \leq q$. For vector values (*i.e.* values consisting of multiple blocks) we use the greek letters α, β with an index for the query, to denote the position within the vector (*e.g.* message M at position α is denoted as M_{i,α_i}). Index α_i runs between $1 \leq \alpha_i \leq \ell_i$, with $|M_i| = \ell_i$.

The tweak values of *COPA* are of the form

$$2^a 3^b 7^c L_i \text{ with } L_i = E_K(0),$$

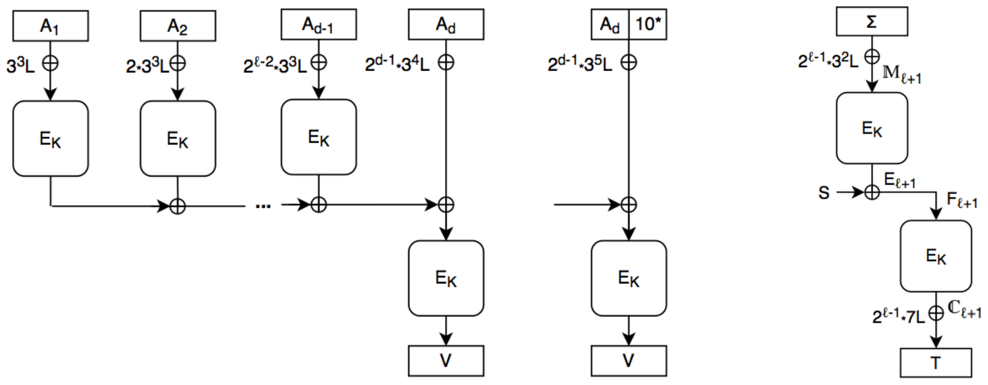


Figure 6.2.: (**left**) Associated Data Processing. (**right**) Tag Generation for COPA.

where the masking values $2^a 3^b 7^c$ are all distinct [Rog04a], with $a, b, c \in \mathbb{N}$. Moreover, we denote tweaks of the top layer as Δ_{i,α_i} and tweaks of the bottom layer as ∇_{i,α_i}. Additionally, we denote the input to the encryption function in the top layer as $\mathbb{M}$ and the output of the encryption function in the bottom layer as $\mathbb{C}$, respectively. Then, we have

$$\mathbb{M}_{i,\alpha_i} = M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}$$

and

$$\mathbb{C}_{i,\alpha_i} = C_{i,\alpha_i} \oplus \nabla_{i,\alpha_i}.$$

Furthermore, the input to the encryption function in the bottom layer is denoted as F and the output of the encryption function in the top layer is denoted as E. This gives us

$$F_{i,\alpha_i} = \begin{cases} E_K^{-1}(C_{i,\alpha_i} \oplus \nabla_{i,\alpha_i}) & \text{if } \alpha_i \le \ell_i \\ E_K^{-1}(T_i \oplus \nabla_{i,\alpha_i}) & \text{if } \alpha_i = \ell_i + 1 \end{cases}$$

and

$$E_{i,\alpha_i} = \begin{cases} L_i \oplus V_i \oplus F_{i,\alpha_i} & \text{if } \alpha_i = 1 \\ F_{i,\alpha_i - 1} \oplus F_{i,\alpha_i} & \text{if } \alpha_i > 1. \end{cases}$$

Finally, we define

$$\Sigma_i = \bigoplus_{\alpha_i = 1}^{\ell_i} M_{i,\alpha_i}$$

as the message checksum for the tag generation. Moreover, for the length given in n-bit blocks we have $|A_i| = d_i$ and $|M_i| = |C_i| = \ell_i$ and $|T_i| = 1$.

Security Bounds of COPA. *COPA* achieves the following security bounds for *privacy*

$$\mathbf{Adv}^{\text{IND-CPA}}_{COPA[E]}(t,q,\sigma,\ell) \leq \frac{39(\sigma+q)^2}{2^n} + \mathbf{Adv}^{\text{SPRP}}_{E}(t,4(\sigma+q)) + \frac{(\ell+1)(q-1)^2}{2^n}$$

and for *integrity*

$$\mathbf{Adv}^{\text{INT-CTXT}}_{COPA[E]}(t,q,\sigma,\ell) \leq \frac{39(\sigma+q)^2}{2^n} + \mathbf{Adv}^{\text{SPRP}}_{E}(t,4(\sigma+q)) + \frac{(\ell+1)(q-1)^2}{2^n} + \frac{2q}{2^n}.$$

Recently, Nandi [Nan15] showed an attack on *COPA*, when XLS is used in case $|M| > n$. Thereby, the security of *COPA* diminishes to

$$\mathbf{Adv}^{\text{IND-PRF}}_{COPA[E]}(t,q,\sigma,\ell) \leq \frac{5.5q^2}{2^n}$$

and

$$\mathbf{Adv}^{\text{IND-CTXT}}_{COPA[E]}(t,q,\sigma,\ell) \leq \frac{2q^3 + 6.5q^2 + 10q + 1}{2^n}.$$

For details about *COPA* we refer to the original paper [ABL+13].

6.3. Nonce-Respecting Security of COPA

In this section, we state our results of the analysis of *COPA*. First, we analyze different variants of $\overline{COPA}$ and give several attacks on these variants. Second, we choose one promising candidate and state a security proof.

Definition 20. *($\overline{COPA}$): $\overline{COPA}$ is a variant of COPA (i.e. one of the candidates of Table 6.1) aimed to achieve beyond birthday bound security.*

To improve the mode *COPA*, we introduce a variant of the original scheme – to achieve beyond birthday bound security. In order to keep the performance of the new variant comparable to *COPA*, the changes must be minimal and not increase the run time of the algorithm by a large factor. Therefore, to increase the security bound we use a nonce in the scheme and restrict the adversary $\mathcal{A}$ to be nonce-respecting. We can place the nonce at several possible positions in the construction of *COPA*. The most interesting candidates are given in Table 6.1. There, we state the position to include the nonce N_i, the security achieved by this variant (attacks are given in Section 6.3) and a comment on the additional operations, if necessary.

Table 6.1.: $\overline{COPA}$ Candidates

Variant	Security	Comment
$C_{i,\alpha_i} \oplus N_i$	Appendix B.1	$\|M_i\| \oplus$ operations
$M_{i,\alpha_i} \oplus N_i$	Appendix B.2	$\|M_i\| \oplus$ operations
$M_{i,\alpha_i} \oplus N_i, C_{i,\alpha_i} \oplus N_i$	Appendix B.3	$2 * \|M_i\| \oplus$ operations
$F_{i,\alpha_o} \oplus N_i$	Appendix B.4	$\|M_i\| \oplus$ operations
$E_{i,\alpha_i} \oplus N_i$	Appendix B.4	$\|M_i\|\|\oplus$ operations
$2^a 3^b 7^c L \oplus N_i$	Appendix B.5	$2 * \|M_i\| + \|A_i\| \oplus$ operations
$2^a 3^b 7^c L\|\|N_i$	Appendix B.6	$\|\Delta\| = \|\nabla\| = 2n$
$\mathbb{M}_{i\alpha_i} \oplus N_i, E_{i,\alpha_i} \oplus N_i$	Appendix B.7	$\|M_i\| \oplus$ operations
N_i incl. in A_i	$2^{n/2}$	Suggested in AES-*COPA*
$V_i \oplus N_i$	$2^{n/2}$	$1 \oplus$ operation
$V_{i,\alpha_i} \oplus 2_i^{\alpha} N_i$	$2^{n/2}$	$\|M_i\| \oplus$ operations + $\|M_i\|$ doublings
$F_{i,\alpha_i} \oplus N_i, \mathbb{C}_{i,\alpha_i} \oplus N_i$	$2^{n/2}$	$2 * \|M_i\| \oplus$ operations
$L_i = E_K(N_i)$	Section 6.3.2	No precomputation of Δ, ∇

In the following, we give an attack on the candidate used in AES-*COPA* and show that including the nonce in such a way doesn't improve the security bound in any way.

Proposition 1. *Let $\mathcal{E}$ denote $\overline{COPA}$, where a nonce N is added anywhere as an associated data block in the* AD*-processing (e.g. prepend or append to $\mathcal{AD}$). Moreover, let $\mathcal{D}$ denote a distinguisher, then the advantage of $\mathcal{D}$ is*

$$\mathbf{Adv}_{\mathcal{E}}^{IND\text{-}CPA}(\mathcal{D}) \leq \frac{\binom{q}{2}}{2^n}$$

where $\mathcal{D}$ runs in time t and makes at most q queries.

Proof. The distinguishing attack here works as follows. Query the oracles ($\mathcal{E}_K$ or \$) with $q_i : (N_i, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{i,\alpha_i}, C_{i,\beta_i}, T_i)$. Furthermore, submit a second query $q_j : (N_i \oplus \delta, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{j,\alpha_i}, C_{j,\beta_i}, T_j)$. Now a collision between $V_i = V_j$ happens with probability $Pr = \binom{q}{2}/2^n$. If we are in the real world, where oracle $\mathcal{E}_K$ represents $\overline{COPA}$ we have a probability for a collision of $C_{i,\alpha_i} = C_{j,\alpha_j}$ and $C_{i,\beta_i} = C_{j,\beta_i}$ with $Pr = 1$, giving a total probability of $Pr = \binom{q}{2}/2^n$. Nevertheless, if we are in the ideal world, where oracle \$ responses for every query with a random string of length $|M_{i,\alpha_i}|$ the probability for a collision between the ciphertexts $C_{i,\alpha_i} = C_{j,\alpha_j}$ and $C_{i,\beta_i} = C_{j,\beta_i}$ is $Pr = 2\binom{q}{2}/2^{2n}$. □

Attacks on Several $\overline{\text{COPA}}$ Candidates. Moreover, also some of the other $\overline{COPA}$ variants show some vulnerabilities. In Appendix B, we give a list of attacks to break the security of several variants as marked in Table 6.1. All these distinguishing attacks are rather simple and break the security of the variant with only two queries. Four of the mentioned variants achieve the same security as *COPA*. Only for one variant, we haven't found any attack – and therefore selected it to perform a security proof, given in the following.

6.3.1. Description of $\overline{\text{COPA}}$

In the following, we give a short description of the candidate $\overline{COPA}$ we have chosen to prove secure beyond the birthday bound. The candidate $\overline{COPA}$ includes the nonce N_i in the tweak values Δ and ∇, where $L_i = E_{K_1}(N_i)$. Moreover, we use three different and independent keys - K_1 for the encryption of the nonce, K_2 for encryptions in the message processing and K_3 for the encryptions in the associated-data processing. Moreover, the associated data is processed via a PRF (*i.e.* $V_i = PRF_{K_3}(N_i, A_i)$). We require, that the adversary $\mathcal{A}$ for $\overline{COPA}$ is nonce-respecting, as contrary to *COPA* where $\mathcal{A}$ is nonce-reusing. Table 6.2 highlights the differences between *COPA* and $\overline{COPA}$.

	COPA	$\overline{COPA}$
$\mathcal{A}$	nonce-reusing	nonce-respecting
L_i	$E_K(0)$	$E_{K_1}(N_i)$
V_i	see Section 6.2.1	$PRF_{K_3}(N_i, A_i)$
K	K	K_1, K_2, K_3

Table 6.2.: Main Differences between *COPA* and $\overline{COPA}$.

Generalization of Security Proof of $\overline{\text{COPA}}$ to COPA. In the *privacy* proof of *COPA*[ABL+13], Andreeva *et al.* make use of the XE and XEX constructions of Rogaway [Rog04a]. There, they introduce dummy masks for the message and associated data processing of *COPA*, in order to replace XE and XEX with random permutations. Furthermore, they show in Lemma 2 [ABL+13], that the PMAC1-like [Rog04a] associated data processing can be replaced by a random function Φ. Our approach for the *privacy* proof of $\overline{COPA}$ follows a similar pattern, where we define the associated data processing of $\overline{COPA}$ as a PRF – which then gets replaced by a random function Φ and furthermore, we replace the encryption functions with random permutations. Therefore, the original *privacy* proof of *COPA* can be seen as a generalization of the proof of $\overline{COPA}$ in Section 6.3.2.

6.3.2. Privacy Proof of $\overline{\text{COPA}}$

In the following, we give the privacy proof for our selected candidate $\overline{COPA}$. Therefore, we use the OAE1 notion for privacy and the SPRP notion defined in Section 5.2.3.

Theorem 4. *Let $\Pi = (\mathcal{E}, \mathcal{E}^{-1})$ denote the candidate $\overline{COPA}$ as defined in Section 6.3.1. Moreover, let $\mathcal{D}$ be a distinguisher making at most q queries to either $\mathcal{E}$ or $\$^{OAE}$ and using time t then,*

$$\textbf{\textit{Adv}}_{\mathcal{E}}^{\textit{IND-CPA}}(\mathcal{D}) \leq \frac{8\binom{\sigma+q}{3}}{(2^n - q)^2} + 2 \cdot \textbf{\textit{Adv}}_{E}^{SPRP}(2(\sigma+q), t') + \textbf{\textit{Adv}}_{F}^{PRF}(q, t').$$

For the proof of Theorem 4 we apply Patarin's Coefficient-H technique defined in [Pat08] and refined by Chen and Steinberger [CS14]. In the following, we give an informal statement on how the proof of Theorem 4 works.

$\overline{COPA}$ uses three different and independent keys – K_1 for the encryption of the nonce $L_i = E_{K_1}(N_i)$, K_2 for the encryptions in the top and bottom layer of the message processing and finally K_3 for the encryptions in the associated data processing. We start by replacing the encryption functions with SPRP's (*i.e.* E_{K_1} with π_1, E_{K_2} with π_2), which adds us the term $2 \cdot \mathbf{Adv}_E^{\text{SPRP}}(2(\sigma + q), t')$ and the PRF processing of the associated data with a random function Φ, adding the term $\mathbf{Adv}_F^{\text{PRF}}(q, t')$. These two value should be close to zero (*i.e.* negligible) in case we use AES or any other good $\mathcal{SE}$-scheme as underlying primitive. Next, we define the good and bad views and evaluate the probabilities of $\$^{\text{OAE}}$ hitting a good and bad view, and $\overline{COPA}$ hitting a good view.

Definition of Views. We define a view ν as

$$\nu = (N_i, AD_i, M_i, V_i, L_i, F_i, C_i, T_i)_{i=1}^{q}$$

where $\sigma = \sum_i^q \ell_i$ denotes the length of length of all ciphertext values in one view, and summed over all queries $1 \le i \le q$

$$\sum_i^q |M_i + \Sigma_i| = \sum_i^q |F_i| = \sum_i^q |C_i + T_i| = \sigma + q.$$

In the ideal world we query the $\$^{\text{OAE}}$ oracle and in the real world $\overline{COPA}$. $\mathcal{A}$ controls the input values N_i, M_i and AD_i in both worlds (ideal and real) for $1 \le i \le q$.

Real World. The values $V_i, L_i, F_{i,\alpha_i}, C_{i,\alpha_i}, T_i$ are generated as defined in Section 6.2.1 and 6.3.1.

Ideal World. Here the values C_{i,α_i}, T_i and V_i are generated independently and uniformly at random. The generation of L_i and F_{i,α_i} is more involved.

◇ $\mathbf{L}_i$. The values L_i are generated uniformly at random, but in such a way that they are all distinct from each other (*i.e.* $\nexists i, j$ *s.t.* $L_i = L_j$).

◇ $\mathbf{F}_{i,\alpha_i}$. The values F_{i,α_i} are generated uniformly at random, but with the condition that if $\mathbb{C}_{i,\alpha_i} = \mathbb{C}_{j,\alpha_j}$ then $F_{i,\alpha_i} = F_{j,\alpha_j}$ and similarly if $\mathbb{M}_{i,\alpha_i} = \mathbb{M}_{j,\alpha_j}$ then $E_{i,\alpha_i} = E_{j,\alpha_j}$. Moreover, if $F_{i,\alpha_i} = \mathbb{M}_{j,\alpha_i}$ then $E_{j,\alpha_j} = \mathbb{C}_{i,\alpha_i}$. All these conditions hold also for the vice versa case.

Definition of Bad Views. A view ν is called *bad* if it is from the set of ν_{bad}. Views are added to ν_{bad} if they fulfill the conditions of a bad transcripts as described below.

Bad Transcripts. We denote the input to the encryption function $E_{K_2}(\cdot)$ as $\mathcal{M}$ and the output as $\mathcal{C}$ such that $E_{K_2}(\mathcal{M}) = \mathcal{C}$. Then we call a view ν a *bad* view, if any of the following conditions holds. We give an informal explanation afterwards.

1. $\exists (i, \alpha_i, \rho_{\alpha_i}), (i, \beta_i, \rho_{\beta_i}), (j, \alpha_j, \rho_{\alpha_j}), (k, \beta_k, \rho_{\beta_k})$
   $s.t.\ \mathcal{M}_{i,\alpha_i,\rho_{\alpha_i}} = \mathcal{M}_{j,\alpha_j,\rho_{\alpha_j}} \wedge \mathcal{M}_{i,\beta_i,\rho_{\beta_i}} = \mathcal{M}_{k,\beta_k,\rho_{\beta_k}}$
2. $\exists (i, \alpha_i, \rho_{\alpha_i}), (i, \beta_i, \rho_{\beta_i}), (j, \alpha_j, \rho_{\alpha_j}), (k, \beta_k, \rho_{\beta_k})$
   $s.t.\ \mathcal{C}_{i,\alpha_i,\rho_{\alpha_i}} = \mathcal{C}_{j,\alpha_j,\rho_{\alpha_j}} \wedge \mathcal{C}_{i,\beta_i,\rho_{\beta_i}} = \mathcal{C}_{k,\beta_k,\rho_{\beta_k}}$

where $1 \le i, j, k \le q$, $1 \le \alpha_i, \beta_i \le \ell_i$, $1 \le \alpha_j \le \ell_j$, $1 \le \beta_k \le \ell_k$ and $\rho_{\alpha_i}, \rho_{\beta_i}, \rho_{\alpha_j}, \rho_{\beta_k} \in \{\mathsf{top}, \mathsf{bot}\}$, where top denotes the top layer and bot the bottom layer, respectively. Furthermore, we limit that the collisions in the i^{th} query must be consecutive. This means that $\beta_i = \alpha_i + 1$.

In words, i, j, k selects the query (q_i, q_j, q_k). $\alpha_i, \beta_i, \alpha_j, \beta_k$ denotes the position (e.g. α_i is between 1 and ℓ_i for query q_i) and $\rho_{\alpha_i}, \rho_{\beta_i}, \rho_{\alpha_j}, \rho_{\beta_k}$ defines if we select a value from the top or bottom layer. Then, the first condition states that we have no consecutive collision in the input values of any encryption with any other input value of another query. Moreover, the second condition states that we have no consecutive collision in the output values of any encryption with any other output value of another query.

Nevertheless, if the F values can be generated randomly, and the bad conditions (1 and 2) does not occur, then there won't ever be a conflict between those values.

Lemma 2. *Bounding the probability of bad transcripts in the ideal world*

$$Pr(\nu(\mathcal{D}^{\$^{OAE}}) \in \nu_{bad}) \le \delta$$

where $\delta = \dfrac{8\binom{\sigma+q}{3}}{(2^n - q)^2}$.

Proof. The intuition behind this proof is to bound the probability of the $\$^{\text{OAE}}$ oracle to hit any view $\nu \in \nu_{bad}$. First, we define which values $\mathcal{M}$ and $\mathcal{C}$ can attain. Then, we analyze with what probability one of the above mentioned bad event occurs – for each of these values. We make a case distinction for each value that $\mathcal{M}$ can attain:

- $\diamond$ $\mathcal{M} = \mathbb{M}_{i,\alpha_i} = M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}$ with $1 \leq \alpha_i \leq \ell_i$ and $1 \leq i \leq q$
- $\diamond$ $\mathcal{M} = \mathbb{M}_{i,\ell_i+1} = \Sigma_i \oplus \Delta_{i,\ell_i+1}$ with $1 \leq i \leq q$
- $\diamond$ $\mathcal{M} = F_{i,\alpha_i}$ with $1 \leq \alpha_i \leq \ell_i + 1$ and $1 \leq i \leq q$

For each of these values we need to analyze with what probability two consecutive collisions within the same query q_i with two other queries q_j, q_k occur. There are four different cases that we need to consider:

- $\diamond$ $(M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}) = (M_{j,\alpha_j} \oplus \Delta_{j,\alpha_j}) \wedge (M_{i,\beta_i} \oplus \Delta_{i,\beta_i}) = (M_{k,\beta_k} \oplus \Delta_{k,\beta_k})$
- $\diamond$ $F_{i,\alpha_i} = F_{j,\alpha_j} \wedge F_{i,\beta_i} = F_{k,\beta_k}$
- $\diamond$ $(M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}) = F_{j,\alpha_j} \wedge (M_{i,\beta_i} \oplus \Delta_{i,\beta_i}) = F_{k,\beta_k}$
- $\diamond$ $F_{i,\alpha_i} = (M_{j,\alpha_j} \oplus \Delta_{j,\alpha_j}) \wedge F_{i,\beta_i} = (M_{k,\beta_k} \oplus \Delta_{k,\beta_k})$

Furthermore, we make a case distinction for each value that $\mathcal{C}$ can attain:

- $\diamond$ $\mathcal{C} = E_{i,\alpha_i} = E_{K_2}(M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i})$ with $1 \leq i \leq \ell_i$
- $\diamond$ $\mathcal{C} = E_{i,\ell_i+1} = E_{K_2}(\Sigma_i \oplus \Delta_{i,\ell_i+1})$
- $\diamond$ $\mathcal{C} = \mathbb{C}_{i,\alpha_i}$ with $1 \leq i \leq \ell_i + 1$

Again for each of these values we analyze with what probability two consecutive collisions within the same query q_i with two other queries q_j, q_k occur. Yet, there are four different cases that we need to consider:

- $\diamond$ $E_{K_2}(M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}) = E_{K_2}(M_{j,\alpha_j} \oplus \Delta_{j,\alpha_j}) \wedge E_{K_2}(M_{i,\beta_i} \oplus \Delta_{i,\beta_i}) =$
  $E_{K_2}(M_{k,\beta_k} \oplus \Delta_{k,\beta_k})$
- $\diamond$ $(C_{i,\alpha_i} \oplus \nabla_{i,\alpha_i}) = (C_{j,\alpha_j} \oplus \nabla_{j,\alpha_j}) \wedge (C_{i,\beta_i} \oplus \nabla_{i,\beta_i}) = (C_{k,\beta_k} \oplus \nabla_{k,\beta_k})$
- $\diamond$ $E_{K_2}(M_{i,\alpha_i} \oplus \Delta_{i,\alpha_i}) = (C_{j,\alpha_j} \oplus \nabla_{j,\beta_j}) \wedge E_{K_2}(M_{i,\beta_i} \oplus \Delta_{i,\beta_i}) = (C_{k,\beta_k} \oplus \nabla_{k,\beta_k})$
- $\diamond$ $(C_{i,\alpha_i} \oplus \nabla_{i,\alpha_i}) = E_{K_2}(M_{j,\alpha_j} \oplus \Delta_{j,\alpha_j}) \wedge (C_{i,\beta_i} \oplus \nabla_{i,\beta_i}) = E_{K_2}(M_{k,\beta_k} \oplus \Delta_{k,\beta_k})$

For every of the above mentioned cases we have to fix three queries q_i, q_j, q_k from a set of $\sigma + q$ possible values. There, we have $\binom{\sigma+q}{3}$ possible combinations to fix these three queries – q_i at position $(\alpha_i, \rho_{\alpha_i})$, q_j at position $(\alpha_j, \rho_{\alpha_j})$ and q_k at position $(\beta_k, \rho_{\beta_k})$. Additionally, the last position $(\beta_i, \rho_{\beta_i})$ in query q_i gets also fixed while choosing q_i. Moreover, the probability for a collision of two $\mathbb{M}, E, F, \mathbb{C}$ values is at most $1/(2^n - q)^2$. Summing all together we get our proposed bound. □

Now consider a good transcript $\nu \in \nu_{good}$. The intuition behind the proofs of Lemma 3 and 4 is to bound the probability that a previously selected view $\nu \in \nu_{good}$ is hit by the $\$^{\mathrm{OAE}}$ oracle in the ideal world and the probability that it is hit by $\overline{COPA}$ in the real world, respectively. Following that, we analyze if the ratio between those probabilities is close to one.

Lemma 3. *Bounding the probability of good transcripts in the ideal world*

$$Pr(\nu(\mathcal{D}^{\$^{OAE}}) = \nu) = \frac{1}{2^{n(\sigma+q)}} * \frac{(2^n - q)!}{2^n!} * \frac{(2^n - \neq_{bottom})!}{2^n!} * \frac{1}{2^{nq}}$$

where $\neq_{bottom} \leq \sigma + q$ and denotes the number of distinct permutations of π_2 in the bottom layer.

Proof. In the ideal world, randomness comes from the ciphertexts C_{i,α_i} and tag values T_i and furthermore from the *dummy* L_i, V_i and F_{i,α_i} values. In the following, we analyze the probability that the $\$^{\mathrm{OAE}}$ oracle hits any of these values for a previously selected good view ν. The oracle $\$^{\mathrm{OAE}}$ generates the ciphertext and tag values independently and uniform random from a set of $\{0,1\}^n$. We have σ ciphertext and q tag values in one view ν. Then, the probability to hit these values is $1/2^{n(\sigma+q)}$, which gives us the first term. The evaluation of the second, third and fourth term is more involved. To evaluate the probability that the random oracle hits the previously selected $\nu \in \nu_{good}$ it is sufficient to compute the fraction of oracles Ω_{comp} that could result in this good view ν. By Ω_{all} we denote the set of all oracles $\mathcal{O}$ for the ideal world. Hence, we have

$$\frac{\Omega_{comp}^{\pi_1}}{\Omega_{all}^{\pi_1}} = \frac{(2^n - q)!}{2^n!}$$

for the evaluation of the q distinct values $L_i = \pi_1(N_i)$, where $\Omega_{all}^{\pi_1} = 2^n!$ is the number of all possible permutations for π_1 and $\Omega_{comp}^{\pi_1} = (2^n - q)!$ the number of compliant oracles after q evaluations of L_i. Moreover, we have

$$\frac{\Omega_{comp}^{\pi_2}}{\Omega_{all}^{\pi_2}} = \frac{(2^n - \neq_{bottom})!}{2^n!}$$

for the evaluation of $\sigma + q$ values of F, where $\Omega_{all}^{\pi_2} = 2^n!$ is the number of possible permutations for π_2. $\Omega_{comp}^{\pi_2} = (2^n - \neq_{bottom})!$ is the number of compliant oracles with $\neq_{bottom}$ the number of distinct permutations of π_2 in the bottom layer. Finally, we have $1/2^{nq}$ for the evaluation of q values of V_i, where each V_i value is generated independently and uniformly random. □

We are left to bound the probability of a good transcript in the real world (*i.e.* $\overline{COPA}$).

Lemma 4. *Bounding the probability of good transcripts in the real world*

$$Pr(\nu(\mathcal{D}^{\overline{COPA}}) = \nu) = \frac{(2^n - q)!}{2^n!} * \frac{(2^n - \neq_{top} - \neq_{bottom})!}{2^n!} * \frac{1}{2^{nq}}$$

where $\neq_{top} \leq \sigma + q$ *and* $\neq_{bottom} \leq \sigma + q$. $\neq_{top}$ *and* $\neq_{bottom}$ *denotes the number of distinct permutations of* π_2 *in the top and bottom layer, respectively.*

Proof. Here we use a similar approach to Lemma 3. In the real world, randomness comes from the permutations π_1, π_2 and π_3. In the following, we analyze with what probability the output values of those permutations is hit by $\overline{COPA}$. To evaluate the probability that $\overline{COPA}$ hits the previously selected $\nu \in \nu_{good}$ it is sufficient to compute the fraction of oracles Ω_{comp} that could result in this good view ν. By Ω_{all} we denote the set of all oracles $\mathcal{O}$ for the real world. Hence, we have

$$\frac{\Omega_{comp}^{\pi_1}}{\Omega_{all}^{\pi_1}} = \frac{(2^n - q)!}{2^n!}$$

for the evaluation of the q distinct values $L_i = \pi_1(N_i)$, where $\Omega_{all}^{\pi_1} = 2^n!$ is the number of all possible permutations for π_1 and $\Omega_{comp}^{\pi_1} = (2^n - q)!$ the number of compliant oracles after q evaluations of L_i. Additionally, we have

$$\frac{\Omega^{\pi_2}_{comp}}{\Omega^{\pi_2}_{all}} = \frac{(2^n - \neq_{top} - \neq_{bottom})!}{2^n!}$$

for the evaluation of $\sigma + q$ values of C, T, where $\Omega^{\pi_2}_{all} = 2^n!$ is the number of possible permutations for π_2. $\Omega^{\pi_2}_{comp} = (2^n - \neq_{top} - \neq_{bottom})!$ is the number of compliant oracles with $\neq_{top}$ the number of distinct permutations of π_2 in the top layer and $\neq_{bottom}$ the number of distinct permutations of π_2 in the bottom layer respectively. Finally, the for the processing of the associated data with the random function Φ we have $1/2^{nq}$ for the evaluation of q values of V_i, where each V_i value is generated independently from Φ. □

The ratio between a good view in the real world to a good view in the ideal world is lower bounded by $1 - \epsilon$. Using the probabilities of Lemma 3 and 4 we get

$$\begin{aligned}
\frac{Pr(\nu(\overline{COPA}) = \nu_{good})}{Pr(\nu(\$^{\mathrm{OAE}}) = \nu_{good})} &= \frac{2^{n(\sigma+q)}(2^n - \neq_{top} - \neq_{bottom})!}{(2^n - \neq_{bottom})!} \\
&\overset{\text{using(6.1)}}{\geq} \frac{2^{n(\sigma+q)}(2^n - \sigma - q - \neq_{bottom})!}{(2^n - \neq_{bottom})!} \\
&\overset{\text{using(6.2)}}{\geq} \frac{2^{n(\sigma+q)}}{(2^n - \neq_{bottom})^{\sigma+q}} \\
&= \left(\frac{2^n}{(2^n - \neq_{bottom})}\right)^{\sigma+q} \\
&\geq 1 \geq 1 - \epsilon
\end{aligned}$$

with $\epsilon = 0$.

$$\neq_{top} \leq \sigma + q \tag{6.1}$$

$$\frac{N!}{(N-x)!} \leq N^x \tag{6.2}$$

The statistical distance $\Delta(\$^{\mathrm{OAE}}, \overline{COPA})$ is now upper bounded by $\delta + \epsilon$.

6.3.3. Integrity of $\overline{\text{COPA}}$

Although, the security bound for *privacy* increases while using a nonce in $\overline{COPA}$, this behavior doesn't apply for *integrity*. In the following, we show an attack on the integrity of $\overline{COPA}$.

Proposition 2. *Let $\Pi = (\mathcal{E}, \mathcal{E}^{-1})$ denote $\overline{COPA}$ as defined in Section 6.3.1. Moreover, let $\mathcal{D}$ be a distinguisher making at most q_e queries to $\mathcal{E}$ and q_d queries to either $\mathcal{E}^{-1}$ or $\perp$ and using time t then,*

$$\boldsymbol{Adv}_{\Pi}^{\text{INT-CTXT}}(\mathcal{D}) \leq \frac{\binom{q}{2}}{2^n}$$

Proof. Our attack works as follows: We fix an associated data block AD, and a message block M. Then, we query the encryption oracle $\mathcal{E}_K$ with queries $q_i : (AD, N_i, M)$, where N_i is always different. We observe, if $T_i = T_j$ for any two queries q_i, q_j. Now $T_i = T_j$ if

$$\exists i, j \text{ s.t. } \mathbb{C}_{i,\ell_i+1} \oplus \nabla_{i,\ell_i+1} = \mathbb{C}_{j,\ell_j+1} \oplus \nabla_{j,\ell_j+1} \tag{6.3}$$

where $\nabla_{i,\ell_i+1} = 2^{\ell_i-1}7E_{K_1}(N_i)$ and $\nabla_{j,\ell_j+1} = 2^{\ell_j-1}7E_{K_1}(N_j)$. The probability for a collision of $T_i = T_j$, that satisfies Equation (6.3) holds with $Pr = 1/2^n$. For q queries, we have $\binom{q}{2}$ choices to satisfy Equation (6.3). Following, that we can query the verification oracle $\mathcal{E}_K^{-1}$ with queries $q_i : (AD, N_i, C_i, T_i)$ and $q_j : (AD, N_j, C_j, T_j)$, where $T_i = T_j$ and get our forgery. □

6.4. Application to AES-COPA

AES-*COPA* is a CAESAR candidate by Andreeva *et al.* [ABL+14b], that uses the AE composition scheme *COPA* and as underlying block cipher AES. The security claims by the designers are in either case – under a nonce-respecting adversary or a nonce-reusing adversary given in Table 6.3.

In AES-*COPA* the nonce – if any – is append to the associated data. Our attack in Section 6.3 shows that appending the nonce to the associated data doesn't improve the security bound. Nevertheless, our results of the scheme

$\overline{COPA}$ can be applied to AES-*COPA* without any restrictions. Then, AES-$\overline{COPA}$ would be secure against chosen plaintext attacks up to approximately 2^{83} AES calls – instead of 2^{64} in the current state.

Table 6.3.: Security Claims of AES-*COPA* [ABL+14b], AES-$\overline{COPA}$, PRØST-*COPA* [KLL+14] and PRØST-$\overline{COPA}$ for $n = \{128, 256\}$.

	AES-*COPA*	AES-$\overline{COPA}$	PRØST-*COPA*	PRØST-$\overline{COPA}$
Confidentiality of plaintext	2^{64}	2^{83}	$2^{64}/2^{128}$	$2^{83}/2^{168}$
Integrity of plaintext	2^{64}	2^{64}	$2^{64}/2^{128}$	$2^{64}/2^{128}$
Integrity of associated data	2^{64}	2^{64}	$2^{64}/2^{128}$	$2^{64}/2^{128}$
Integrity of PMN	2^{64}	2^{64}	-	-
Security against key recovery	2^{128}	-	-	-
Security against tag guessing	2^{128}	-	-	-

† For *integrity* of $\overline{COPA}$ we conjecture that the security bound of *COPA* carries over.

6.5. Application to PRØST-COPA

PRØST is a CAESAR candidate by Kavun *et al.* [KLL+14]. PRØST per se, defines a strong permutation. To achieve AE, the PRØST-family uses the constructions schemes OTR [Min14], *COPA* [ABL+13] and APE [ABB+15], with PRØST as underlying primitive – either directly or via the single-keyed Even-Mansour scheme. In this thesis, we only consider PRØST with the mode *COPA*.

Description of PRØST-COPA-n. PRØST in the mode *COPA* is called PRØST-*COPA*-n, where $2n$ represents the permutation size of the underlying permutation PRØST. n can be either 128 or 256 bits – resulting in a security of 2^{64} or 2^{128}, respectively. PRØST-*COPA*-n uses the single-keyed Even-Mansour construction to construct the block cipher needed in *COPA*. It is defined on a $2n$-bit key K such that $\widetilde{P}_{n,K} : \{0,1\}^{2n} \times \{0,1\}^{2n} \to \{0,1\}^{2n}$ or $\widetilde{P}_{n,K}(x) = K \oplus P_n(x \oplus K)$. In *COPA* the function $E_K(x)$ is then replaced by $\widetilde{P}_{n,K}(x)$.

Application of $\overline{\text{COPA}}$ to Prøst-COPA-n. Again, PRØST-*COPA-n* appends the nonce to the associated data, such as $\overline{\text{Aes-copa}}$ – which doesn't improve the security bound. Likewise, our results on $\overline{COPA}$ can be applied to Prøst increasing the security bound from 2^{64} to approximately 2^{83}, or from 2^{128} to approximately 2^{168}, respectively. Our results are stated in Table 6.3.

6.6. Deoxys/Joltik/KIASU

Deoxys, Joltik and KIASU are CAESAR candidates designed by Jean *et al.* [JNP14b, JNP14c, JNP14d]. They build upon the TWEAKEY framework [JNP14a] and the resulting blockciphers Deoxys-BC, Joltik-BC and KIASU-BC. Furthermore, to achieve AE they use the modes *COPA* and OCB [RK14]. In this thesis only the *COPA* variants are considered.

6.6.1. Description of Deoxys/Joltik/KIASU

Deoxys, Joltik and KIASU in the nonce-reusing mode *COPA* are called Deoxys$^=$, Joltik$^=$ and KIASU$^=$. The structure of the candidates is the same, only the underlying tweakable blockcipher differs. They take as input an arbitrary but finite-length associated data A and message M, that are separated into n-bit blocks (*i.e.* $A = (A_1 \ldots A_d)$ and $M = (M_1 \ldots M_\ell)$, where each $|A_i| = |M_i| = n$). For associated data and messages that are not a multiple of the block length the last M/A block is padded using the $10^\star$ *padding*. It outputs a ciphertext C, split into $\ell \times n$-bit blocks $C_1 \ldots C_\ell$ and a tag T of length $|T| = n$. The encryption process is illustrated in Figure 6.3 and 6.4.

Notation of Deoxys$^=$, Joltik$^=$ and KIASU$^=$. The tweak values are of the form

$$(domain\ separator\ ||\ N\ ||\ ctr)$$

where we denote Δ_{i,α_i} as a top layer tweak and ∇_{i,α_i} as a bottom layer tweak value for query q_i at position α_i. Furthermore, we denote again the output of

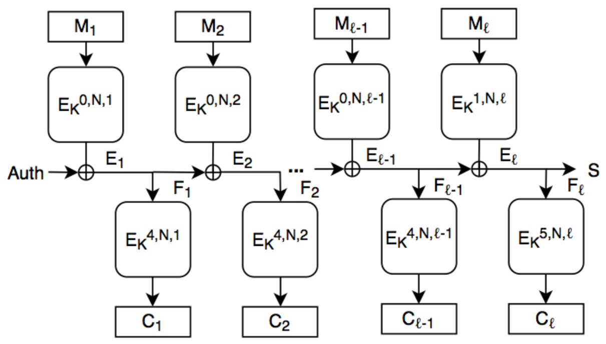


Figure 6.3.: Message Processing for Deoxys$^=$, Joltik$^=$ and KIASU$^=$.

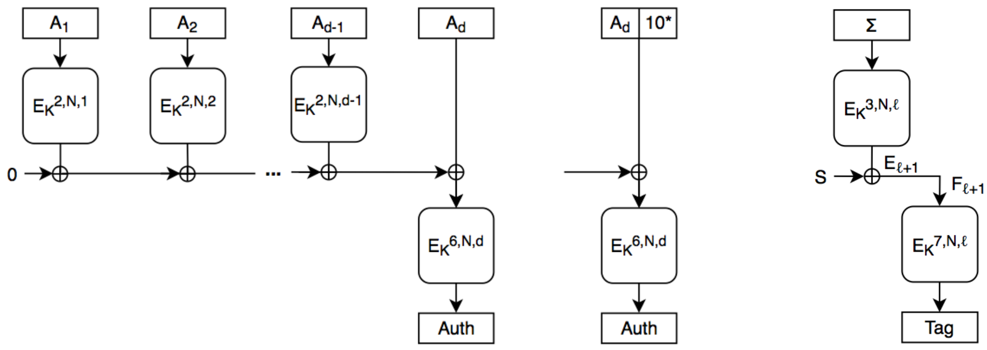


Figure 6.4.: (**left**) Associated Data Processing. (**right**) Tag Generation for Deoxys$^=$, Joltik$^=$ and KIASU$^=$.

the encryption function in the top layer as E and the input to the encryption function in the bottom layer as F. Thus, we have

$$E_{i,\alpha_i} = E_K^{\Delta_{i,\alpha_i}}(M_{i,\alpha_i}) \text{ and}$$

$$F_{i,\alpha_i} = D_K^{\nabla_{i,\alpha_i}}(C_{i,\alpha_i}).$$

Finally, we define

$$\Sigma_i = \bigoplus_{\alpha_i=1}^{\ell_i} M_{i,\alpha_i}$$

as the message checksum for the tag generation. Moreover, for the length given in n-bit blocks we have $|A_i| = d_i$ and $|M_i| = |C_i| = \ell_i$ and $|T_i| = 1$.

6.6.2. Privacy Proof of Deoxys$^=$/Joltik$^=$/Kiasu$^=$

For the proof of Deoxys$^=$/Joltik$^=$/Kiasu$^=$ we use the OAE1 notion for privacy and the INT-CTXT notion for integrity and furthermore the $\widetilde{SPRP}$ notion defined in Section 5.2.2 and 5.2.3.

Theorem 5. *Let $\mathcal{E}$ denote Deoxys, Joltik or Kiasu in the mode COPA with $E_K(\cdot,\cdot)$ the `TWEAKEY` construction. Furthermore, let $\mathcal{D}$ be a distinguisher, that runs in time t and makes at most q queries of total length σ to either $\mathcal{E}$ or $\$^{OAE}$, then*

$$\boldsymbol{Adv}_{\mathcal{E}}^{IND\text{-}CPA}(\mathcal{D}) \leq \boldsymbol{Adv}_{\texttt{TWEAKEY}}^{\widetilde{SPRP}}(\mathcal{D}_1)$$

where $\mathcal{D}_1$ is another distinguisher that runs in similar time $t' \approx t$ and makes at least 2σ queries.

We start by giving an informal statement on how the proof of Theorem 5 works. First, we show that the tweak values Δ_i, ∇_i are unique for every query q_i. Next, we replace the `TWEAKEY` encryption functions by $\widetilde{SPRP}$'s and show that Deoxys$^=$, Jolitk$^=$ and Kiasu$^=$ achieves optimal security.

Lemma 5. *The tweak values Δ_{i,α_i} and ∇_{i,α_i} of Deoxys$^=$, Jolitk$^=$ and Kiasu$^=$ are unique for each associated data AD_{i,α_o} and message block M_{i,α_i} at any position α_i for any query q_i up to $2^{|N|}$ queries.*

Proof. The tweak values consists of the concatenation of a domain separator, a unique nonce and a counter (*i.e.* $\{\Delta_{i,\alpha_i}, \nabla_{i,\alpha_i}\} = (domain\ separator \,||\, N \,||\, \alpha_i)$). The probability for a collision of two tweak values is $Pr(\Delta_{i,\alpha_i} = \Delta_{j,\alpha_j}) = Pr(\nabla_{i,\alpha_i} = \nabla_{j,\alpha_j}) = Pr(\Delta_{i,\alpha_i} = \nabla_{j,\alpha_j}) = 0$ and for $Pr(\Delta_{i,\alpha_i} = \Delta_{j,\alpha_i}) = Pr(\nabla_{i,\alpha_i} = \nabla_{j,\alpha_i}) = 2^{-|N|}$, because after $2^{|N|}$ the nonces repeat and we only have to look at the same message/associated data block.

The nonces N_i are public known and under the full control of the adversary. Nevertheless, it is not possible to create collisions between different message/associated data blocks, for a fixed nonce N_i in a query q_i, due to the fact of different domain separators and counters (*i.e.* message ctr/associated data ctr). □

As the first step we replace the TWEAKEY encryption functions $\widetilde{E}_K^{\Delta_{i,\alpha_i}}$ and $\widetilde{E}_K^{\nabla_{i,\alpha_i}}$ with $\widetilde{SPRP}$'s. Such an replacement costs us $\mathbf{Adv}_{\mathtt{TWEAKEY}}^{\widetilde{SPRP}}(2\sigma, t \approx t')$, where we get for each input $2\widetilde{E}$ function calls, that take time $t' \approx t$, resulting from the additional operations (*e.g.* xor of intermediate values).

Lemma 6. *Let $\mathcal{E}$ denote Deoxys, Joltik or Kiasu in the mode COPA with $\widetilde{\pi}_K^{\Delta_{i,\alpha_i}}$ and $\widetilde{\pi}_K^{\Delta_{i,\alpha_i}}$. Furthermore, let $\mathcal{D}$ be a distinguisher which runs in time t and makes at most q queries to $\mathcal{E}$ or $\$^{OAE}$, then*

$$\mathbf{Adv}_{\mathcal{E}}^{\text{IND-CPA}}(q, \sigma, t) \leq 0 + \mathbf{Adv}_{\mathtt{TWEAKEY}}^{\widetilde{SPRP}}(2\sigma, t')$$

Proof. Without loss of generality we only need to look on encryptions in the bottom layer of Deoxys$^{=}$, Jolitk$^{=}$ or Kiasu$^{=}$, due to the fact that $\mathcal{D}$ can only observe ciphertexts C_{i,α_i} and tag value T_i as response to its queries q_i.

Now every $\widetilde{\pi}_K^{\nabla_{i,\alpha_i}}$ returns a unique value from the set of $\{0,1\}^n$ for each input block F_{i,α_i} for every query q_i. In other words, every different (i, α_i) results in a new value. In the top layer $\widetilde{\pi}_K^{\Delta_{i,\alpha_i}}$ generates random E_{i,α_i} values, which again generates random F_{i,α_i} values. Then, a collision between $C_{i,\alpha_i} = C_{j,\alpha_j}$ occurs with probability $Pr = \binom{q}{2}/2^n$, which is the same as for the $\$^{\text{OAE}}$ oracle. Therefore, $\mathcal{D}$ has a zero advantage to distinguish between $\mathcal{E}$ and $\$^{\text{OAE}}$. □

The results of Lemma 6 proves Theorem 5.

6.6.3. Integrity Proof of Deoxys$^{=}$/Joltik$^{=}$/Kiasu$^{=}$

We use the INT-CTXT notion for integrity. Our goal is to show that an adversary $\mathcal{A}$ has only a negligible advantage to produce a forgery. Then, the INT-CTXT-advantage of $\mathcal{A}$ is defined as

$$\begin{aligned}\mathbf{Adv}_{\mathcal{E}}^{\text{INT-CTXT}}(\mathcal{A}) &= Pr[\mathcal{A}^{\mathcal{E}_K^{\pm 1}(\cdot,\cdot,\cdot)} \text{ forges}] \\ &= \left|Pr[\mathcal{A}^{\mathcal{E}_K(\cdot,\cdot,\cdot),\mathcal{E}_K^{-1}(\cdot,\cdot,\cdot)} \Rightarrow 1] - Pr[\mathcal{A}^{\mathcal{E}_K(\cdot,\cdot,\cdot),\perp(\cdot,\cdot,\cdot)} \Rightarrow 1]\right|\end{aligned}$$

where the probabilities are taken from $K \xleftarrow{\$} \mathcal{K}$. Although, we are in a nonce-respecting setting – $\mathcal{A}$ can reuse the nonce in a decryption query to generate a forgery. Moreover, $\mathcal{A}$ is restricted to never submit to the decryption oracle any trivial query, where it has previously obtained the answer from the encryption oracle and vice versa. Nevertheless, $\mathcal{A}$ has full control over the values N_i, A_i, M_i, C_i, T_i. It now tries to find a tag $T_j = T_i$, where any of the values N_i, A_i, M_i, C_i is new. A forgery is successful, if the decryption oracle returns something other than $\perp$ for a query q_i with new values.

Theorem 6. *Let $(\mathcal{E}, \mathcal{E}^{-1})$ denote Deoxys$^{=}$, Joltik$^{=}$ or Kiasu$^{=}$, with $E_K(\cdot,\cdot)$ the* `TWEAKEY` *construction. Moreover, let $\mathcal{D}$ be a distinguisher making at most q_e queries to $\mathcal{E}$ and q_d queries to either $\mathcal{E}^{-1}$ or $\perp$ and using time t then,*

$$\boldsymbol{Adv}_{\mathcal{E}}^{INT\text{-}CTXT}(\mathcal{D}) \leq \frac{3 \cdot q_d}{2^n}$$

Proof. We succeed with our forgery attempt, if we can submit a query q_i to the decryption/verification oracle $\mathcal{E}_K^{-1}$ such that it returns something other than $\perp$ (*i.e.* the message M_i). Then, for each query q_i $\mathcal{A}$ must fulfill the following equation to produce a valid tag

$$F'_{i,\ell_i+1} = F_{i,\ell_i+1}$$

where $F'_{i,\ell_i+1} = \widetilde{\pi}^{-1}(T_i)$ and $F_{i,\ell_i+1} = \bigoplus_i(\widetilde{\pi}(M_{i,\alpha_i})) \oplus auth_i \oplus \widetilde{\pi}(\Sigma_i)$. $\widetilde{\pi}^{-1}(\cdot)$ is the inverse of $\widetilde{\pi}_K^{\nabla_{i,\ell_i+1}}$ and $\widetilde{\pi}(\cdot) = \{\widetilde{\pi}_K^{\nabla_{i,\alpha_i}}, \widetilde{\pi}_K^{\Delta_{i,\ell_i+1}}\}$. Now every $\widetilde{\pi}_K^{\nabla_{i,\alpha_i}}$ returns an unique value from the set of $\{0,1\}^n$ for each input block F_{i,α_i} for every query q_i, where we always have a new nonce N_i. For a decryption query we may reuse nonce N_i. Then, if we consider one decryption query

- ◇ N_i is new: The decryption $\widetilde{\pi}^{-1}(T_i)$ returns a random value, where $Pr(F'_{i,\ell_i+1} = F_{i,\ell_i+1}) = 1/2^n$.
- ◇ N_i is old: Since the adversary is nonce-respecting, there exist exactly one encryption query with that nonce N_i. Now, the probability that $F'_{i,\ell_i+1} = F_{i,\ell_i+1}$ for this encryption and decryption query is $Pr = 1/2^n$. Otherwise, F_{i,ℓ_i+1} is new from the set $1/(2^n - 1)$.

Then, the probability is at most $\max\{1/2^n, 1/2^n + 1/(2^n - 1)\} \leq 3/2^n$, where summed over all decryption queries q_d we have $3 \cdot q_d/2^n$ □

6.6.4. Security Bounds for Deoxys$^=$/Joltik$^=$/Kiasu$^=$

In Table 6.4, we give the resulting security bounds for our proofs of Deoxys$^=$, Joltik$^=$ and Kiasu$^=$. Jean *et al.* [JNP14b, JNP14c, JNP14d] conjectured, that Deoxys$^=$, Joltik$^=$ and Kiasu$^=$ achieve full security in the nonce-respecting setting. With our results we can support this claim.

Table 6.4.: Security Bounds for Deoxys$^=$, Joltik$^=$ and Kiasu$^=$ for $n = \{128\}$.

	Deoxys$^=$	Joltik$^=$	Kiasu$^=$
Confidentiality of plaintext	2^{128}	2^{128}	2^{128}
Integrity of plaintext	2^{125}	2^{125}	2^{125}
Integrity of associated data	2^{125}	2^{125}	2^{125}
Integrity of PMN	2^{125}	2^{125}	2^{125}

7 Conclusion

In this thesis, we have analyzed $\overline{COPA}$, a variant of the AE composition scheme *COPA*. *COPA* was presented at Asiacrypt 2013 by Andreeva *et al.* and features the first fully parallelizable online authenticated encryption scheme. Moreover, Andreeva *et al.* provided a security proof in the nonce-ignoring setting with security up to the birthday bound. In order to be applicable to a wide range of applications, we introduce in this thesis a variant of *COPA*, which we prove to be secure beyond the birthday bound under the restriction of a nonce-respecting adversary.

The recently announced CAESAR competition tries to identify a portfolio of secure and robust authenticated encryption schemes with associated data. *COPA* is a AE composition scheme, used in several candidate submissions. With the results achieved in this thesis, we serve in favor of the idea behind the CAESAR to provide well analyzed and therefore secure and robust final candidates.

In this thesis, we analyzed several candidates for their capability to achieve higher security, than the original *COPA* scheme. Andreeva *et al.* proved in their original paper [ABL+13], that with the scheme *COPA* it is only possible to achieve birthday bound security - for both *privacy* and *integrity*. Now, to achieve security beyond the birthday bound, we were forced to limit the adversary to be nonce-respecting. Therefore, we were able to identify several possible candidates (see Table 6.1). Nevertheless, we were able to completely break most of them, using simple attacks. Furthermore, more advanced candidates provided only up to the birthday bound security. But beside all these problems, we were able to identify one candidate scheme, that after further analysis, we could prove beyond birthday bound security. For the proof of security, we used a specific technique – already used in several other security proofs – Patatin's coefficient-H technique. The results of this proof, could be verbatim used in the candidates AES-*COPA* and PRØST, in mode *COPA*. Therefore, we were able to increase the security bound for IND-CPA from 2^{64} to 2^{83} encryption calls for AES-*COPA* and from $2^{64}(2^{128})$ to $2^{83}(2^{168})$ encryption calls, respectively for PRØST-*COPA*.

The candidates Deoxys, Joltik and KIASU use the AE composition scheme *COPA* in a slightly different variation – such that our proof of $\overline{COPA}$ doesn't apply. Therefore, we provide in this thesis an additional *privacy* and *integrity* proof considering *COPA* is this slightly different setting. The designers of Deoxys, Joltik and KIASU – Jean *et al.* already conjectured beyond birthday bound security [JNP14b, JNP14c, JNP14d] – for an nonce-respecting adversary. Within the proof in this thesis, we give first results that this conjecture can be indeed true.

Since, the work of this thesis was limited in time we can provide with several ideas to further improve these bounds and encourage fellow cryptographers to use these results. Since an adversary can reuse nonces for decryption queries in the INT-CTXT and IND-CCA notion the *integrity* bound of $\overline{COPA}$ can not exceed the birthday bound. Due to the limited time, we could not proof a bound for *integrity* of $\overline{COPA}$. Moreover, one can probably improve the security bound for *privacy* by using the constructions proposed by Mennink at FSE 2015 [Men15]. In addition, one can refine our conditions for bad views to provide with a higher *privacy* bound.

Bibliography

[ABB$^+$15] E. Andreeva, B. Bilgin, A. Bogdanov, A. Luykx, B. Mennink, N. Mouha, and K. Yasuda. APE: Authenticated Permutation-Based Encryption for Lightweight Cryptography. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 168–186. Springer, 2015.

[ABL$^+$13] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, E. Tischhauser, and K. Yasuda. Parallelizable and Authenticated Online Ciphers. Cryptology ePrint Archive, Report 2013/790, 2013.

[ABL$^+$14a] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, N. Mouha, and K. Yasuda. How to Securely Release Unverified Plaintext in Authenticated Encryption. In *Advances in Cryptology – ASIACRYPT 2014*, Lecture Notes in Computer Science, pages 105–125. Springer, 2014.

[ABL$^+$14b] E. Andreeva, A. Bogdanov, A. Luykx, B. Mennink, E. Tischhauser, and K. Yasuda. AES-COPA. CAESAR submission. `http://competitions.cr.yp.to/round1/aescopav1.pdf`, 2014.

[AD] E. Andreeva and X. Dutoit. Visualization of CAESAR candidates. `http://homes.esat.kuleuven.be/~eandreev/caesarviz`.

[AFL$^+$15a] F. Abed, C. Forler, E. List, S. Lucks, and J. Wenzel. Don't Panic! The Cryptographers Guide to Robust Authenticated (On-Line) Encryption. `https://www.uni-weimar.de/fileadmin/user/`

fak/medien/professuren/Mediensicherheit/Research/Drafts/nonce-misuse-oae.pdf, 2015.

[AFL15b] F. Abed, C. Forler, and S. Lucks. General Overview of the First-Round CAESAR Candidates for Authenticated Encryption. Cryptology ePrint Archive, Report 2014/792, 2015.

[AKL$^+$] F. Abed, S. Koelbl, M. Lauridsen, C. Rechberger, and T. Tiessen. Authenticated Encryption Zoo. https://aezoo.compute.dtu.dk/doku.php?id=ae_zoo.

[BDPVA11] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. The Keccak reference. Submission to NIST (Round 3), 2011.

[Ber] D. J. Bernstein. CAESAR - Competition for Authenticated Encryption: Security, Applicability, and Robustness. http://competitions.cr.yp.to/caesar.html.

[Ber08] D. J. Bernstein. The Salsa20 Family of Stream Ciphers. In *New Stream Cipher Designs*, Lecture Notes in Computer Science, pages 84–97. Springer, 2008.

[BN00] M. Bellare and C. Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In *Advances in Cryptology — ASIACRYPT 2000*, Lecture Notes in Computer Science, pages 531–545. Springer, 2000.

[BR93] M. Bellare and P. Rogaway. Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security - CCS 1993*, pages 62–73. ACM, 1993.

[BR02] J. Black and P. Rogaway. A Block-Cipher Mode of Operation for Parallelizable Message Authentication. In *Advances in Cryptology — EUROCRYPT 2002*, Lecture Notes in Computer Science, pages 384–397. Springer, 2002.

[BR04] M. Bellare and P. Rogaway. The Game-Playing Technique, 2004.

[BRW04] M. Bellare, P. Rogaway, and D. Wagner. The EAX Mode of Operation. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 389–407. Springer, 2004.

[CBK97] R. Canetti, M. Bellare, and H. Krawczyk. HMAC: Keyed-Hashing for Message Authentication, 1997.

[CMN+14] S. Cogliani, D. Maimut, D. Naccache, R. Portella do Canto, R. Reyhanitabar, S. Vaudenay, and D. Vizár. Offset Merkle-Damgrd (OMD). CAESAR submission. `http://competitions.cr.yp.to/round1/omdv10.pdf`, 2014.

[CS14] S. Chen and J. Steinberger. Tight Security Bounds for Key-Alternating Ciphers. In *Advances in Cryptology – EUROCRYPT 2014*, Lecture Notes in Computer Science, pages 327–350. Springer, 2014.

[CW79] C. Carter and M. Wegman. Universal Classes of Hash Functions. *Journal of Computer and System Sciences*, pages 143 – 154, 1979.

[Die08] T. Dierks. The Transport Layer Security (TLS) Protocol. RFC 5246, IETF, 2008.

[DKA+14] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson. The Matter of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference - IMC 2014*, pages 475–488. ACM, 2014.

[DKS12] O. Dunkelman, N. Keller, and A. Shamir. Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In *Advances in Cryptology - EUROCRYPT 2012*, Lecture Notes in Computer Science, pages 336–354. Springer, 2012.

[DMK14] T. Duong, B. Moeller, and K. Kotowicz. This POODLE Bites: Exploiting The SSL3.0 Fallback, 2014.

[DR98] J. Daemen and V. Rijmen. AES Proposal: Rijndael, 1998.

[DR11] T. Duong and J. Rizzo. Here Come the Xor Ninjas. BEAST Attack, 2011.

[Dwo04] M. J. Dworkin. SP 800-38c. Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality. Technical report, 2004.

[EM93] S. Even and Y. Mansour. A Construction of a Cipher from a Single Pseudorandom Permutation. In *Advances in Cryptology — ASIACRYPT 1991*, Lecture Notes in Computer Science, pages 210–224. Springer, 1993.

[FFL12] E. Fleischmann, C. Forler, and S. Lucks. McOE: A Family of Almost Foolproof On-Line Authenticated Encryption Schemes. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 196–215. Springer, 2012.

[FLS$^+$10] N. Ferguson, S. Lucks, B. Schneier, D. Whiting, M. Bellare, T. Kohno, J. Callas, and J. Walker. The Skein Hash Function Family. Submission to NIST (Round 3), 2010.

[FTC$^+$77] H. Feistel, W. Tuchman, D. Coppersmith, A. Konheim, C. Meyer, M. Matyas, R. Adler, E. Grossman, B. Notz, L. Smitz, and B. Tuckerman. Data Encryption Standard. In *In FIPS PUB 46, Federal Information Processing Standards Publication*, pages 2–27, 1977.

[GKM$^+$11] P. Gauravaram, L. Knudsen, K. Matusiewicz, F. Mendel, C. Rechberger, M. Schläffer, and S. Thomsen. Grøstl – a SHA-3 candidate. Submission to NIST (Round 3), 2011.

[HCJ02] S. Halevi, D. Coppersmith, and C. Jutla. Scream: A Software-Efficient Stream Cipher. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 195–209. Springer, 2002.

[Hel80] M. Hellman. A Cryptanalytic Time-Memory Trade-Off. *IEEE Transactions on Information Theory*, pages 401–406, 1980.

[HRRV15] V. Hoang, R. Reyhanitabar, P. Rogaway, and D. Vizar. Online Authenticated-Encryption and its Nonce-Reuse Misuse-Resistance. Cryptology ePrint Archive, Report 2015/189, 2015.

[JLM14] P. Jovanovic, A. Luykx, and B. Mennink. Beyond $2^{c/2}$ Security in Sponge-Based Authenticated Encryption Modes. In *Advances in Cryptology – ASIACRYPT 2014*, Lecture Notes in Computer Science, pages 85–104. Springer, 2014.

[JNP14a] J. Jean, I. Nikolić, and T. Peyrin. Tweaks and Keys for Block Ciphers: The TWEAKEY Framework. In *Advances in Cryptology – ASIACRYPT 2014*, Lecture Notes in Computer Science, pages 274–288. Springer, 2014.

[JNP14b] J. Jeremy, I. Nikolic, and T. Peryin. Deoxys. CAESAR submission. `http://competitions.cr.yp.to/round1/deoxysv1.pdf`, 2014.

[JNP14c] J. Jeremy, I. Nikolic, and T. Peryin. Joltik. CAESAR submission. `http://competitions.cr.yp.to/round1/joltikv1.pdf`, 2014.

[JNP14d] J. Jeremy, I. Nikolic, and T. Peryin. KIASU. CAESAR submission. `http://competitions.cr.yp.to/round1/kiasuv1.pdf`, 2014.

[Jut00] C. Jutla. Encryption Modes with Almost Free Message Integrity. Cryptology ePrint Archive, Report 2000/039, 2000.

[Kel02] J. Kelsey. Compression and Information Leakage of Plaintext. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 263–276. Springer, 2002.

[Ken05] S. Kent. IP Encapsulating Security Payload (ESP). RFC 4303, IETF, 2005.

[KL07] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2007.

[KLL$^+$14] E. Kavun, M. Lauridsen, G. Leander, C. Rechberger, P. Schwabe, and T. Yalcin. Prøst. CAESAR submission. `http://competitions.cr.yp.to/round1/proestv11.pdf`, 2014.

[Kor07] T. Korvetz. VMAC: Message Authentication Code using Universal Hashing, 2007.

[KVW04] T. Kohno, J. Viega, and D. Whiting. CWC: A High-Performance Conventional Authenticated Encryption Mode. 2004.

[LR88] M. Luby and C. Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. pages 373–386, 1988.

[LRW02] M. Liskov, R. Rivest, and D. Wagner. Tweakable Block Ciphers. In *Advances in Cryptology — CRYPTO 2002*, Lecture Notes in Computer Science, pages 31–46. Springer, 2002.

[Lu15] J. Lu. On the Security of the COPA and Marble Authenticated Encryption Algorithms against (Almost) Universal Forgery Attack. Cryptology ePrint Archive, Report 2015/079, 2015.

[Mat94] M. Matsui. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology*, EUROCRYPT 1993, pages 386–397. Springer, 1994.

[Men15] B. Mennink. Optimally Secure Tweakable Blockciphers. Cryptology ePrint Archive, Report 2015/363, 2015.

[Min14] K. Minematsu. Parallelizable Rate-1 Authenticated Encryption from Pseudorandom Functions. In *Advances in Cryptology – EUROCRYPT 2014*, Lecture Notes in Computer Science, pages 275–292. Springer, 2014.

[MV04] D. McGrew and J. Viega. The Galois/Counter Mode of Operation (GCM). Submission to NIST Modes of Operation Process, 2004.

[MVO96] A. Menezes, S. Vanstone, and P. Oorschot. *Handbook of Applied Cryptography*. CRC Press, Inc., 1996.

[Nan14] M. Nandi. XLS is Not a Strong Pseudorandom Permutation. In *Advances in Cryptology – ASIACRYPT 2014*, Lecture Notes in Computer Science, pages 478–490. Springer, 2014.

[Nan15] M. Nandi. Revisiting Security Claims of XLS and COPA. Cryptology ePrint Archive, Report 2015/444, 2015.

[NRS14] C. Namprempre, P. Rogaway, and T. Shrimpton. Reconsidering Generic Composition. In *Advances in Cryptology – EUROCRYPT 2014*, Lecture Notes in Computer Science, pages 257–274. Springer, 2014.

[Pat08] J. Patarin. The "Coefficients H" Technique. In *Selected Areas in Cryptography – SAC 2008*, pages 328–345, 2008.

[Riv87] R. Rivest. Rc4, 1987.

[RK14] P. Rogaway and T. Krovetz. OCB. CAESAR submission. `http://competitions.cr.yp.to/round1/ocbv1.pdf`, 2014.

[Rog04a] P. Rogaway. Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. In *Advances in Cryptology - ASIACRYPT 2004*, Lecture Notes in Computer Science, pages 16–31. Springer, 2004.

[Rog04b] P. Rogaway. Nonce-Based Symmetric Encryption. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 348–358. Springer, 2004.

[RR07] T. Ristenpart and P. Rogaway. How to Enrich the Message Space of a Cipher. In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 101–118. Springer, 2007.

[RS06] P. Rogaway and T. Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In *Advances in Cryptology - EUROCRYPT 2006*, Lecture Notes in Computer Science, pages 373–390. Springer, 2006.

[RV14] R. Reyhanitabar and D. Vizar. Careful with Misuse Resistance of online AEAD. `https://groups.google.com/forum/#!topic/crypto-competitions/o5uMRvi6L74`, 2014.

[RZ11] P. Rogaway and H. Zhang. Online Ciphers from Tweakable Blockciphers. In *Topics in Cryptology – CT-RSA 2011*, Lecture Notes in Computer Science, pages 237–249. Springer, 2011.

[Sch94] B. Schneier. Description of a New Variable-length Key, 64-bit Block Cipher (Blowfish). In *Fast Software Encryption*, Lecture Notes in Computer Science, pages 191–204. Springer, 1994.

[Sch98] R. Schroeppel. Hasty Pudding Cipher Spezification, 1998.

[Shr04] T. Shrimpton. A Characterization of Authenticated-Encryption as a Form of Chosen-Ciphertext Security. 2004.

[STA$^+$14] Y. Sasaki, Y. Todo, K. Aoki, Y. Naito, T. Sugawara, Y. Murakami, M. Matsui, and S. Hirose. Minalpher v1. CAESAR submission. `http://competitions.cr.yp.to/round1/minalpherv1.pdf`, 2014.

[Ylo06] T. Ylonen. The Secure Shell (SSH) Transport Layer Protocol. RFC 4253, IETF, 2006.

A

Algorithms for COPA

In this chapter, we define algorithms COPA-Encrypt and COPA-Decrypt, with their procedures AD for associated data processing, Tag for the tag generation, Verify for the tag verification, $\mathcal{E}$ the encryption function of the construction $COPE$ and $\mathcal{E}^{-1}$ the decryption function of the construction $COPE$.

We assume all variables to be initialized with zero. Furthermore, we assume M, AD are multiples of the blocksize n. For the handling of arbitrary message/associated data length we refer to [ABL+13].

Algorithm A.1: COPA-ENCRYPT

```
procedure COPA-ENCRYPT()
  Input: $E, K, A, M$
  Output: $C, T$
11 :    $L \leftarrow E_K(0)$
12 :    $V \leftarrow AD(E, K, L, A)$
13 :    $V \leftarrow V \oplus L$
14 :    $\{C, S\} \leftarrow \mathcal{E}(E, K, V, L, M)$
15 :    $T \leftarrow Tag(E, K, S, L, M)$
16 :    return $C, T$

procedure AD()
  Input: $E, K, L, A$
  Output: $V$
21 :    $\Delta \leftarrow 3^3 L$
22 :    $A_1 || A_2 \cdots A_d \leftarrow A$, with $|A_i| = n$
          for $1 \leq i \leq d$
23 :    for $i = 1, \ldots, d-1$ do
24 :        $Y \leftarrow Y \oplus E_K(A_i \oplus \Delta)$
25 :        $\Delta \leftarrow 2\Delta$
26 :    $V \leftarrow E_K(Y \oplus A_d \oplus 3\Delta)$
27 :    return $V$

procedure TAG()
  Input: $E, K, L, S, M$
  Output: $T$
31 :    $M_1 || M_2 \cdots M_\ell \leftarrow M$, with $|M_i| = n$
          for $1 \leq i \leq \ell$
32 :    for $i = 1, \ldots, \ell$ do
33 :        $\Sigma \leftarrow \Sigma \oplus M_i$
34 :    $\Sigma \leftarrow E_K(\Sigma \oplus 2^{\ell-1} 3^2 L)$
35 :    $T \leftarrow E_K(\Sigma \oplus S) \oplus 2^{\ell-1} 7L$
36 :    return $T$

procedure $\mathcal{E}$()
  Input: $E, K, V, L, M$
  Output: $C, S$
41 :    $\Delta \leftarrow 3L; \nabla \leftarrow 2L$
42 :    $M_1 || M_2 \cdots M_\ell \leftarrow M$, with $|M_i| = n$
          for $1 \leq i \leq \ell$
43 :    for $i = 1, \ldots, \ell$ do
44 :        $V_i \leftarrow E_K(M_i \oplus \Delta) \oplus V_{i-1}$
45 :        $C_i \leftarrow E_K(V_i) \oplus \nabla$
46 :        $\Delta \leftarrow 2\Delta; \nabla \leftarrow 2\nabla$
47 :    return $C, V_\ell$
```

Algorithm A.2: COPA-DECRYPT

```
procedure COPA-DECRYPT()
  Input: $E, E^{-1}, K, A, C, T$
  Output: $M$ or $\perp$
11 :   $L \leftarrow E_K(0)$
12 :   $V \leftarrow AD(E, K, L, A)$
13 :   $V \leftarrow V \oplus L$
14 :   $\{M, S\} \leftarrow \mathcal{E}^{-1}(E^{-1}, K, V, L, C)$
15 :   return $Verify(E, E^{-1}, K, S, L, M, T)$

procedure AD()
  Input: $E, K, L, A$
  Output: $V$
21 :   $\Delta \leftarrow 3^3 L$
22 :   $A_1 || A_2 \cdots A_d \leftarrow A$, with $|A_i| = n$
         for $1 \leq i \leq d$
23 :   for $i = 1, \ldots, d-1$ do
24 :       $Y \leftarrow Y \oplus E_K(A_i \oplus \Delta)$
25 :       $\Delta \leftarrow 2\Delta$
26 :   $V \leftarrow E_K(Y \oplus A_d \oplus 3\Delta)$
27 :   return $V$

procedure VERIFY()
  Input: $E, E^{-1}, K, L, S, M, T$
  Output: $M$ or $\perp$
31 :   $M_1 || M_2 \cdots M_\ell \leftarrow M$, with $|M_i| = n$
         for $1 \leq i \leq \ell$
32 :   for $i = 1, \ldots, \ell$ do
33 :       $\Sigma \leftarrow \Sigma \oplus M_i$
34 :   if $S \oplus E_K(\Sigma \oplus 2^{\ell-1} 3^2 L) =$
         $E_K^{-1}(T \oplus 2^{\ell-1} 7L)$ then
35 :       return $M$
36 :   return $\perp$

procedure $\mathcal{E}^{-1}$()
  Input: $E^{-1}, K, V, L, C$
  Output: $M, S$
41 :   $\Delta \leftarrow 3L; \nabla \leftarrow 2L$
42 :   $C_1 || C_2 \cdots C_\ell \leftarrow C$, with $|C_i| = n$
         for $1 \leq i \leq \ell$
43 :   for $i = 1, \ldots, \ell$ do
44 :       $V_i \leftarrow E_K^{-1}(C_i \oplus \nabla)$
45 :       $M_i \leftarrow E_K^{-1}(V_i \oplus V_{i-1}) \oplus \Delta$
46 :       $\Delta \leftarrow 2\Delta; \nabla \leftarrow 2\nabla$
47 :   return $M, V_\ell$
```

B
Attacks on $\overline{\text{COPA}}$ Candidates

In this chapter, we give some sample queries for attacks on the $\overline{COPA}$ candidates defined in Table 6.1. The following attacks are all distinguishing attacks between $\mathcal{E}_K$, which represents the $\overline{COPA}$ candidate and the oracle \$, which responses with a random string of length $|M_{i,\alpha_i}|$ for every new query q_i. Furthermore, consider for these attacks that $\alpha_i = 1$, always the first position in the query q_i and $\beta_i = \alpha_i + 1$. Moreover, δ represents any difference.

Then, submit to $\mathcal{E}_K$ or \$

- ◇ **Attack 1.** $q_i : (N_i, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{i,\alpha_i}, C_{i,\beta_i}, T_i)$ and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{j,\alpha_i}, C_{j,\beta_i}, T_j)$
  Verify if $C_{i,\alpha_i} \oplus C_{i,\beta_i} = C_{j,\alpha_i} \oplus C_{j,\beta_i}$.
- ◇ **Attack 2.** $q_i : (N_i, M_{i,\alpha_i})$ to receive (C_{i,α_i}, T_i) and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i} \oplus \delta)$ to receive (C_{j,α_i}, T_j)
  Verify if $C_{i,\alpha_i} = C_{j,\alpha_i}$.

- ◇ **Attack 3.** $q_i : (N_i, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{i,\alpha_i}, C_{i,\beta_i}, T_i)$ and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i} \oplus \delta, M_{i,\beta_i} \oplus \delta)$ to receive $(C_{j,\alpha_i}, C_{j,\beta_i}, T_j)$
  Verify if $C_{i,\alpha_i} \oplus C_{i,\beta_i} = C_{j,\alpha_i} \oplus C_{j,\beta_i}$.
- ◇ **Attack 4.** $q_i : (N_i, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{i,\alpha_i}, C_{i,\beta_i}, T_i)$ and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{j,\alpha_i}, C_{j,\beta_i}, T_j)$
  Verify if $C_{i,\beta_i} = C_{j,\beta_i}$.
- ◇ **Attack 5.** $q_i : (N_i, M_{i,\alpha_i} \oplus N_i)$ to receive (C_{i,α_i}, T_i) and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i} \oplus N_i \oplus \delta)$ to receive (C_{j,α_i}, T_j)
  Verify if $C_{i,\alpha_i} \oplus N_i = C_{j,\alpha_i} \oplus N_i \oplus \delta$.
- ◇ **Attack 6.** If $|M_{i,\alpha_i}| = 2n$
  $q_i : (N_i, M_{i,\alpha_i}||N_i)$ to receive (C_{i,α_i}, T_i) and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i}||N_i \oplus \delta)$ to receive (C_{j,α_i}, T_j)
  Verify if $C_{i,\alpha_i} \oplus (0^n||N_i) = C_{j,\alpha_i} \oplus (0^n||N_i) \oplus \delta$.
- ◇ **Attack 7.** $q_i : (N_i, M_{i,\alpha_i}, M_{i,\beta_i})$ to receive $(C_{i,\alpha_i}, C_{i,\beta_i}, T_i)$ and
  $q_j : (N_i \oplus \delta, M_{i,\alpha_i} \oplus \delta, M_{i,\beta_i} \oplus \delta)$ to receive $(C_{j,\alpha_i}, C_{j,\beta_i}, T_j)$
  Verify if $C_{i,\beta_i} = C_{j,\beta_i}$.