

# **The Design and Analysis of Symmetric Cryptosystems**

Gordon Procter

Thesis submitted to the University of London  
for the degree of Doctor of Philosophy

Information Security Group  
Royal Holloway, University of London

2015

# Declaration

---

These doctoral studies were conducted under the supervision of Prof. Carlos Cid.

The work presented in this thesis is the result of original research carried out by myself, in collaboration with others, whilst enrolled in the Department of Information Security as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree or award in any other university or educational establishment.

Gordon Procter  
August 2015

# Acknowledgements

---

First and foremost, I wish to thank my supervisor Carlos Cid. Throughout this PhD, he has provided guidance, support, and advice. He has encouraged me explore the topics and areas that I find interesting and always had an interesting observation to make.

Many thanks are due to the co-authors that I have had the privilege to work with. I've enjoyed productive collaborations with Pooya Farshim, Loïc Ferreira, and Matt Robshaw. I also owe Jean Paul Degabriele thanks, for his contributions to both my education and my research.

The ECRYPT project has provided great opportunities for me to join a wonderful community of European researchers working on symmetric cryptography. I am very grateful for the events they have organised (with their generous travel funding) and to everyone within that community.

The department at Royal Holloway has been an excellent venue for the last 4 years: sincere thanks to everyone that has made it so. My fellow PhD students have kept me going with humour, squash, and coffee. Kenny and Bertram have provided motivation, opportunities, and interesting ideas. Susan and James deserve a special mention, for the sarcasm and the essential life lessons.

Thanks to my parents, for supporting me in everything I do.

Finally, thanks to Rachael, for putting up with it all.

# Abstract

---

Message authentication schemes built from universal hash functions are commonly used for fast and secure message authentication. By studying universal hash functions based on polynomial evaluation, we identify some properties which arise from the underlying algebraic structure. As a result, we are able to describe a general forgery attack against the related message authentication schemes, as well as providing a common description of all known attacks against such schemes, and greatly expanding the number of known weak keys.

Iterated Even–Mansour ciphers are also popular and we initiate the theoretical study of these ciphers’ security against related-key attacks. The simplest one-round Even–Mansour cipher is shown to achieve a non-trivial level of related-key security. However, offsetting keys by constants is not included in this result; two rounds suffice to reach that level of security under chosen-plaintext attacks and three rounds boosts security to resist chosen-ciphertext attacks.

Tweakable block ciphers are a generalisation of block ciphers that take an additional input (the tweak) in order to provide an efficient alternative to re-keying the cipher. We analyse the security reduction given for CLRW2, a method for constructing a tweakable block cipher from a (conventional) block cipher and a universal hash function. Having identified an error in the proof, we provide a revised proof with a new bound.

Finally, we study the security of two schemes that have been proposed for standardisation. The first is a composition of Bernstein’s ChaCha20 and Poly1305, as proposed for use in IETF protocols as an authenticated encryption scheme; the second is an ultra-lightweight RFID authentication protocol proposed as part of ISO/IEC 29167. We conclude that the first is a secure authenticated encryption scheme, while the second is catastrophically broken by algebraic attacks.

# Contents

---

<b>1</b>	<b>Introduction</b>	<b>10</b>
1.1	Motivation and context . . . . .	10
1.2	Thesis structure . . . . .	13
<b>2</b>	<b>Preliminaries</b>	<b>16</b>
2.1	Notation . . . . .	16
2.2	Cryptographic background . . . . .	17
2.2.1	Pseudorandom functions . . . . .	17
2.2.2	Block ciphers . . . . .	18
2.2.3	Cryptographic permutations . . . . .	21
2.2.4	Symmetric encryption schemes . . . . .	22
2.2.5	Tweakable block ciphers . . . . .	25
2.2.6	Message authentication codes . . . . .	27
2.2.7	Universal hash functions . . . . .	29
2.2.8	Authenticated encryption . . . . .	32
2.3	Security models . . . . .	35
2.3.1	Weak keys . . . . .	35
2.3.2	Related-key attacks . . . . .	36
2.3.3	Indifferentiability . . . . .	38
2.3.4	Algebraic cryptanalysis . . . . .	41
<b>3</b>	<b>Polynomial-based hash functions</b>	<b>42</b>
3.1	Introduction . . . . .	43
3.2	Prior work . . . . .	44
3.2.1	Polynomial-evaluation-based hash functions . . . . .	44
3.2.2	Security analyses . . . . .	48
3.3	Algebraic structure and its implications . . . . .	52
3.3.1	A generalised forgery attack . . . . .	52
3.3.2	Malleability . . . . .	55
3.3.3	Length extension . . . . .	56
3.3.4	Key recovery . . . . .	57
3.3.5	Choosing polynomials . . . . .	58
3.4	Algebraic structure of previous attacks . . . . .	60
3.4.1	Ferguson’s short tag attack . . . . .	60
3.4.2	Joux’s forbidden attack . . . . .	61
3.4.3	Handschuh and Preneel’s attacks . . . . .	61
3.4.4	Saarinen’s cycling attacks . . . . .	62
3.5	Weak keys . . . . .	63

## CONTENTS

---

3.5.1	Handschuh and Preneel’s weak key . . . . .	63
3.5.2	Saarinen’s weak keys . . . . .	64
3.5.3	New weak-key classes . . . . .	64
3.6	GCM with short multiplications . . . . .	66
3.6.1	Introduction . . . . .	66
3.6.2	Attacks . . . . .	67
3.7	Square Hash . . . . .	70
3.7.1	Introduction . . . . .	70
3.7.2	Weak-key classes . . . . .	71
3.8	Discussion . . . . .	73
3.8.1	Choice of fields . . . . .	74
3.8.2	Length extension . . . . .	74
3.8.3	Malleability . . . . .	75
3.8.4	Weak keys . . . . .	75
<b>4</b>	<b>The related-key security of iterated Even–Mansour ciphers</b>	<b>77</b>
4.1	Introduction . . . . .	78
4.2	Preliminaries . . . . .	80
4.2.1	Notation . . . . .	80
4.2.2	The Even–Mansour ciphers . . . . .	81
4.2.3	Security Analyses . . . . .	82
4.2.4	Cryptanalysis . . . . .	83
4.3	The RKA security of $EM^\pi[1, 1]$ . . . . .	83
4.3.1	Restricting RKD sets . . . . .	84
4.3.2	Sufficiency of the conditions . . . . .	88
4.4	The RKCPA security of $EM^\pi[1, 1, 1]$ . . . . .	99
4.4.1	Weakening the conditions . . . . .	100
4.4.2	Sufficiency of the conditions . . . . .	102
4.4.3	$\Phi^\oplus$ -RKA Security . . . . .	110
4.4.4	A $\Phi^\oplus$ -RKCCA attack on $EM^\pi[1, 1, 1]$ . . . . .	113
4.5	The RKCCA security of $EM^\pi[1, 1, 1, 1]$ . . . . .	114
4.5.1	Attacking $EM^\pi[\kappa]$ for $\kappa \neq [1, 1, 1, 1]$ . . . . .	114
4.5.2	The security of $EM^\pi[1, 1, 1, 1]$ . . . . .	115
4.6	Discussion . . . . .	125
<b>5</b>	<b>The CLRW2 tweakable block cipher</b>	<b>127</b>
5.1	Introduction . . . . .	127
5.2	Preliminaries . . . . .	129
5.2.1	Description of CLRW2 . . . . .	130
5.2.2	Notation . . . . .	130
5.3	Proof summary . . . . .	131
5.3.1	Partitioning the output set . . . . .	131
5.3.2	Structure of the games . . . . .	133
5.3.3	Games 4 and 5 . . . . .	135
5.4	The flaw in the proof . . . . .	139
5.4.1	Flawed sampling methods . . . . .	139
5.4.2	Causing an inversion . . . . .	140

5.5	A revised proof . . . . .	141
5.6	A limitation of this proof technique . . . . .	150
<b>6</b>	<b>Standardisation and cryptography</b>	<b>152</b>
6.1	Introduction . . . . .	152
6.2	The composition of ChaCha20 and Poly1305 . . . . .	154
6.2.1	Notation . . . . .	155
6.2.2	Description of the algorithms . . . . .	156
6.2.3	Security model . . . . .	158
6.2.4	Our result . . . . .	159
6.3	Algebraic cryptanalysis and ISO/IEC 29167-15 . . . . .	164
6.3.1	The standardisation landscape for UHF RFID . . . . .	164
6.3.2	The first version of ISO/IEC 29167-15 . . . . .	166
6.3.3	The working draft of ISO/IEC 29167-15 . . . . .	168
6.3.4	The first committee draft of ISO/IEC 29167-15 . . . . .	170
6.3.5	The final committee draft of ISO/IEC 29167-15 . . . . .	175
6.3.6	Other insecure variants of ISO/IEC 29167-15 . . . . .	181
6.3.7	Results and discussion . . . . .	184
6.4	Discussion . . . . .	185
	<b>Bibliography</b>	<b>187</b>

# Abbreviations

---

3GPP	3rd Generation Partnership Project
AE	Authenticated Encryption
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
ANSI	American National Standards Institute
ARX	(modular) Addition, Rotation, Xor
ATM	Automated Teller Machine
AXU	Almost XOR Universal
BC	Block Cipher
CBC	Cipher Block Chaining
CCA	Chosen Ciphertext Attack
CD	Committee Draft
CF	Claw Freeness
CF1	First-order Claw Freeness
CFB	Cipher Feedback
CFRG	Crypto Forum Research Group
CLRW2	Chained LRW2
CPA	Chosen Plaintext Attack
CRC	Cyclic Redundancy Check
CTR	Counter
CTXT	Ciphertext
DES	Data Encryption Standard
DIS	Draft International Standard
ECB	Electronic Codebook
EM	Even–Mansour
EMV	Europay, MasterCard, and Visa
ENISA	European Union Agency for Network and Information Security
FDIS	Final Draft International Standard
FIPS	Federal Information Processing Standard
GCD	Greatest Common Divisor
GCHQ	Government Communication Headquarters
GCM	Galois/Counter Mode
GCM/2 <sup>+</sup>	Galois/Counter Mode with short multiplications
HF	High Frequency
HMAC	Hash-based Message Authentication Code
IDEA	International Data Encryption Algorithm
IEC	International Electrotechnical Commission

IETF	Internet Engineering Task Force
IND	Indistinguishability
INT	Integrity
IPsec	Internet Protocol Security
IRTF	Internet Research Task Force
ISO	International Organization for Standardization
IV	Initialisation Vector
LRW2	Tweakable Block Cipher proposed by Liskov, Rivest, and Wagner
MAC	Message Authentication Code
MMH	Multilinear Modular Hashing
NIST	National Institute of Standards and Technology
OCB	Offset Codebook
OFB	Output Feedback
OUP	Output Unpredictable
OUP1	First-order Output Unpredictable
PMAC	Parallelisable Message Authentication Code
PRF	Pseudorandom Function
PRG	Pseudorandom Generator
PRP	Pseudorandom Permutation
QI	Query Independence
QI1	First-order Query Independence
RFC	Request for Comments
RFID	Radio Frequency Identification
RKA	Related-key Attack
RKCCA	Related-key Chosen Ciphertext Attack
RKCPA	Related-key Chosen Plaintext Attack
RKD	Related-key-deriving
SAT	Boolean Satisfiability
SES	Symmetric Encryption Scheme
SGCM	Sophie Germaine Counter Mode
SHA	Secure Hash Algorithm
SPRP	Strong Pseudorandom Permutation
SSH	Secure Shell
SSL	Secure Socket Layer
TBC	Tweakable Block Cipher
TLS	Transport Layer Security
TPRP	Tweaked Pseudorandom Permutation
TSPRP	Tweaked Strong Pseudorandom Permutation
UHF	Ultra High Frequency
WD	Working Draft
XCB	Extended Codebook
XCF	Xor Claw Freeness
XE	Xor-Encrypt
XEX	Xor-Encrypt-Xor
XQI	Xor Query Independence
XTS	XEX-based Tweaked-codebook Mode with Ciphertext Stealing

# Introduction

---

## Contents

---

1.1	Motivation and context . . . . .	10
1.2	Thesis structure . . . . .	13

---

*This chapter gives an overview of the thesis. We provide the motivation for our research and give a broad overview of the wider context. In this chapter, we also present the overall structure and main contributions of this thesis.*

## 1.1 Motivation and context

The long-standing aim of cryptography is to facilitate secure communication between a number of parties. The definition of secure communication has broadened over time: historically confidentiality was the prime concern (indeed this is still the perception that many have of cryptography) whereas now data integrity, authentication, and non-repudiation are also recognised as important factors. Similarly, the range of problems for which cryptographic solutions have been considered has increased to include more exotic notions such as providing zero-knowledge proofs and the public verification of calculations that have been outsourced to an untrusted party. Symmetric cryptography aims to solve some of these problems, under the assumption that users share some information (the key) not known to an adversary.

Prior to the 1970s, all cryptography was symmetric and most occurred in a military context (typically via mechanical, rotor-based enciphering machines such as

## 1.1 Motivation and context

---

Scherbius’s Enigma [247]). Two developments changed this: the proposal of public-key cryptography and the standardisation of the Data Encryption Standard (DES). The first public descriptions of public-key cryptography came from Diffie and Hellman [95] and Rivest, Shamir, and Adleman [232]. Ellis and Cocks are now acknowledged as the first to envisage this concept [114]; their research was not declassified until 1997 as they both worked at the Government Communication Headquarters (GCHQ). By separating the public and private keys, public-key cryptography offers two main advantages over symmetric cryptography: simpler key exchange and a greater flexibility within the schemes. Despite this, most data is still encrypted symmetrically: public-key algorithms are more computationally taxing than symmetric algorithms of a comparable strength and the advantages of both systems can be realised by using a public-key algorithm to exchange a symmetric key.

In the intervening decades, symmetric cryptography has found many practical applications. For example, it is used widely across the internet, automated teller machine (ATM) networks, and in mobile phones; it is set to become more ubiquitous as the Internet of Things [156], SmartMeters [171], and radio frequency identification (RFID) tags [153] become more widely used. The many different parties involved in the specification and deployment of these systems and the desire for interoperability necessitates some level of standardisation. Perhaps the best known example of standardised cryptography is the Advanced Encryption Standard (AES): a four-year competition concluded with Rijndael being selected as the replacement for DES. Since then AES has seen widespread use (e.g. [138, 181, 245, 229, 180, 265]). There are many standardisation bodies, four particularly relevant examples for this thesis are the Internet Engineering Task Force (IETF), the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC), and the National Institute of Standards and Technology (NIST).

The first main theme in this thesis is that the cryptography studied is practical: the majority of the schemes studied are either deployed and used, or have been proposed as alternatives for existing deployed schemes. The polynomial-based hash functions studied in Chapter 3 are central to the AES–GCM and ChaCha20–Poly1305 cipher suites in the Transport Layer Security (TLS) protocol, which are used for approximately 40% of TLS connections [179]. In Chapter 4, we study the iterated Even–Mansour design for block ciphers, the principles of which are used in well-known

## 1.1 Motivation and context

---

block ciphers such as AES, Present, and LED. Tweakable block ciphers, which we study in Chapter 5, have become a popular primitive from which to design modes of operation and in particular for disk encryption applications; examples can be found as submissions to the CAESAR competition [79], such as OCB [170] and Minalpher [246], and as standardised disk encryption modes, such as XTS [101]. The final chapter of this thesis is dedicated to studying schemes that have been proposed specifically for standardisation: ChaCha20 and Poly1305 for use within TLS [211], and Crypto-xor as an RFID authentication protocol submitted to ISO/IEC [140].

The papers of Goldwasser and Micali (e.g. [127]) represent a significant milestone in the development of modern theoretical cryptography, laying the foundations for the reductionist security paradigm. In 1917, Vernam [268] described what has come to be known as the one-time pad, which Shannon [249] showed in 1949 to provide Perfect Secrecy—that is, even against adversaries with unlimited computing power, the plaintext remains completely hidden. However, schemes providing perfect secrecy have severe limitations, such as infeasibly long keys. Goldwasser and Micali’s idea [128] was to base the security of a scheme on an (assumed) intractable problem and provide a proof that an adversary breaking the scheme could also solve the problem. While schemes providing perfect secrecy give strong security guarantees, this reductionist paradigm allows for the analysis of practical schemes.

Bellare and Rogaway [18, 234] extend this reductionist approach to ‘*practice-oriented provable security*’. Recognising that the schemes typically constructed in the reductionist approach were too inefficient to be widely used, Bellare and Rogaway popularised the analysis of schemes constructed from finite pseudorandom function families and block ciphers.

Practice-oriented provable security has also encouraged concrete security analyses. In contrast to the methodology of Goldwasser and Micali (who considered asymptotic results and polynomial reducibility), concrete security gives more precise statements of the security of a scheme: for example, an adversary attacking a particular encryption scheme (who runs for a certain, bounded length of time) will have an advantage bounded by some probability, which is expressed in terms of the number of plaintext/ciphertext pairs that they observe and a measure of the security of the underlying block cipher.

## 1.2 Thesis structure

---

The aim of this approach is a community working with precise definitions of security, careful parametrisation of adversaries' capabilities, and thorough exposition of the assumptions made. These ideas have formalised the folklore techniques of building more complex cryptographic schemes from simpler ones.

For an attack against a scheme with a security reduction, several possible interpretations are observed throughout this thesis (noting security reductions' controversy [164] and limitations [37]). The attack may succeed with a lower probability than is described by the security reduction; we will see an example of this in Chapter 3. It may represent a failure of the adversarial model to accurately capture how the scheme is being used; related-key attacks (which we study in Chapter 4) are an example of a strengthened adversarial model, aiming to prevent this issue. There may be an error in the security reduction; this is the case for the scheme studied in Chapter 5. Finally, the attack might include an observation that invalidates the assumption underlying the security of the scheme; in Chapter 6 we study the security of the primitive underlying the Crypto-xor scheme [140].

## 1.2 Thesis structure

CHAPTER 2. In this chapter we introduce the notation, cryptographic primitives, and security notions used throughout this thesis.

CHAPTER 3. Families of hash functions based on polynomial evaluation are often used in the design of message authentication codes and authenticated encryption schemes. Universal hash function families are an attractive choice for this application, as they come with information-theoretic guarantees about their security.

In this chapter, we identify some properties of a universal hash function family that arise from the underlying algebraic structure. We go on to describe a general forgery attack and provide a common description of all known attacks against a related message authentication scheme. We also greatly expand the number of known weak keys for schemes built from this family of hash functions and provide some analysis of schemes based on another family of hash functions.

## 1.2 Thesis structure

---

The work described in this chapter is joint work with Carlos Cid and is published as [226] and [227]; it is also available at [225].

CHAPTER 4. Unless one wishes to gain information-theoretic security, via the use of a scheme such as the one-time pad (along with its well known key-management issues), the security of any scheme will be based on some computational assumption; this assumption critically underpins the security of the scheme. Some uses of block ciphers require security under related keys and if one wishes to instantiate such a scheme with a particular block cipher, then some reasoning as to why that block cipher provides this level of security is required. Almost all modern block cipher proposals will include an initial security analysis, however a more attractive strategy is to design a block cipher in such a way as to obtain theoretical guarantees about its security.

In this chapter we discuss key-alternating ciphers and the theoretical security guarantees that they offer with respect to related-key attacks. We show that the simplest one-round Even–Mansour cipher achieves a non-trivial level of related-key security. Although this does not include the practically relevant case of offsetting keys by constants, two rounds suffice to reach this level under chosen-plaintext attacks and three rounds boosts security to resist chosen-ciphertext attacks.

The work is joint work with Pooya Farshim and is published as [109]; it is also available at [108]. The publication that this chapter is based on also includes a result on the relationship between indistinguishability and RKA security. This result is Farshim’s and so is omitted here; all other contributions are joint.

CHAPTER 5. Tweakable block ciphers are a generalisation of block ciphers that take an additional input (the ‘*tweak*’) in order to provide an efficient alternative to re-keying the cipher. Using tweakable block ciphers to construct higher-level schemes (such as symmetric encryption schemes or authenticated encryption schemes) has become a popular design approach and it is possible to build a tweakable block cipher from a conventional block cipher. Many such constructions come with a security reduction, however the careful verification of such a proof is an essential part of establishing their security.

## 1.2 Thesis structure

---

In this chapter we describe an error in the proof for the CLRW2 construction by Landecker, Shrimpton, and Terashima [176] which uses a conventional block cipher and a family of hash functions to realise a tweakable block cipher. We are able to resolve the issue and give a new bound for the security of CLRW2. This work is available at [223].

CHAPTER 6. It is desirable for standardised cryptography to use primitives and schemes that are well established and have received considerable analysis, however this is not always possible (for example, heavily constrained devices may not have the processing power necessary). In this chapter, we study the security of two schemes that have been proposed for standardisation. The first is a novel composition [211] of Bernstein’s ChaCha20 [35] and Poly1305 [34], as proposed for use in IETF protocols as an authenticated encryption scheme; the second is an ultra-lightweight RFID authentication protocol proposed as part of ISO/IEC 29167 [140].

Our conclusions are that the first is a secure authenticated encryption scheme, while the second is catastrophically broken by algebraic attacks. Algebraic attacks have been successfully applied to a number of other ciphers and authentication protocols. In these attacks the scheme is represented as a set of equations with variables corresponding to secret values (such as bits of the key) expressed in terms of known values (such as observed ciphertext); solving this system of equations corresponds to recovering the secret values.

The analysis of ChaCha20 and Poly1305 is available at [224]. The work on ISO/IEC 29167 is joint with Carlos Cid, Loïc Ferreira, and Matthew Robshaw; it is published as [74].

# Preliminaries

---

## Contents

<b>2.1</b>	<b>Notation . . . . .</b>	<b>16</b>
<b>2.2</b>	<b>Cryptographic background . . . . .</b>	<b>17</b>
<b>2.3</b>	<b>Security models . . . . .</b>	<b>35</b>

---

*This chapter introduces the notation used throughout this thesis and describes the necessary theoretical foundations.*

## 2.1 Notation

We write  $x \leftarrow y$  for the action of assigning the value  $y$  to the variable  $x$ . For a set  $\mathcal{X}$ , we write  $x \leftarrow_s \mathcal{X}$  to represent an element  $x$  being uniformly sampled from  $\mathcal{X}$ . If  $\mathcal{A}$  is a probabilistic algorithm we write  $y \leftarrow_s \mathcal{A}(x_1, \dots, x_n)$  for the action of running  $\mathcal{A}$  on inputs  $x_1, \dots, x_n$  with randomly chosen coins and assigning the result to  $y$ .

For an arbitrary bitstring  $x \in \{0, 1\}^*$  we use  $|x|$  to denote its length. The set of bitstrings with length at most  $\ell$  is denoted  $\{0, 1\}^{\leq \ell}$ . The truncation to  $n$  bits of a string  $x$  will be represented by  $\text{trunc}_n(x)$ . For a set  $S \subseteq \{0, 1\}^n$  and an element  $x \in \{0, 1\}^n$  we define  $S \oplus x = \{s \oplus x : s \in S\}$ . For a bit string  $x$ , we denote the bitwise complement of  $x$  by  $\bar{x}$ . For a bitstring  $x \in \{0, 1\}^{mn}$ , we parse this as  $x_1 || \dots || x_m$  where  $|x_i| = n$  for each  $i$ . If  $x \in \{0, 1\}^*$  with  $|x| = mn + s$ , then we define a partial final block:  $x_{m+1} \in \{0, 1\}^s$ . Throughout this thesis, we will assume that all schemes operate on bitstrings.

## 2.2 Cryptographic background

---

For an algorithm  $\mathcal{A}$ , access to an oracle  $\mathcal{O}$  (or a tuple of oracles  $\mathcal{O} = (\mathcal{O}_1, \dots, \mathcal{O}_t)$ ) will be denoted  $\mathcal{A}^{\mathcal{O}}$ . The number of queries made by  $\mathcal{A}$  to  $\mathcal{O}$  will be denoted by  $q$ ; if  $\mathcal{A}$  has access to more than one oracle, subscripts will be used to distinguish between calls to each of the oracles. In games, all Boolean flags are initialised to `false` and arrays are initially undefined at every point.

A finite field will be denoted by  $\mathbb{K}$  unless the order of the field has particular relevance, in which case it will be denoted by  $\mathbb{F}_{p^r}$  with  $|\mathbb{F}_{p^r}| = p^r$ . The multiplicative group of a field  $\mathbb{K}$  will be denoted by  $\mathbb{K}^*$ . The set of integers will be denoted by  $\mathbb{Z}$  and the set of integers modulo  $p$  by  $\mathbb{Z}_p$ . The set of natural numbers will be denoted by  $\mathbb{N}$ . We use  $\xi(p)$  to represent a Bernoulli random variable that is 1 with probability  $p$  and 0 with probability  $1 - p$ . We let  $[n] = \{1, \dots, n\}$ .

## 2.2 Cryptographic background

In this section, we describe the syntax and security of the cryptographic notions that will be used throughout this thesis. Additionally, some background information on the history and development of these concepts is given.

### 2.2.1 Pseudorandom functions

**BACKGROUND.** A pseudorandom function (PRF) family is a set of functions with the property that, when one is chosen at random, its input-output behaviour cannot be distinguished from a truly random function (by some computationally restricted adversary). This notion was first described by Goldwasser and Micali (under the name ‘*poly-random collections*’) [126]; throughout this thesis we will focus on the finite variant of this property described by Bellare, Killian, and Rogaway [25, 26], as it is better aligned with concrete security analyses.

The idea of pseudorandom functions has been widely adopted as a method with which to build and analyse symmetric-key primitives, particularly block ciphers, encryption schemes, and message authentication codes (e.g. [125, 188, 24, 28, 157]).

## 2.2 Cryptographic background

---

$\text{PRF}_{\mathcal{F}, \mathcal{A}}:$ $b \leftarrow_s \{0, 1\}; k \leftarrow_s \mathcal{K}_{\mathcal{F}}$ $\text{urf} \leftarrow_s \text{Func}(\{0, 1\}^{n_a}, \{0, 1\}^{n_b})$ $b' \leftarrow_s \mathcal{A}^{\text{PRF}}$ $\text{Return } (b' = b)$	$\text{PRF}(X):$ $\text{If } b = 0 \text{ Return urf}(X)$ $\text{Return prf}(k, X)$
---	---

Figure 2.1: Game defining the PRF security of a function family  $\mathcal{F}$ .

**SYNTAX.** A pseudorandom function family between  $\{0, 1\}^{n_a}$  and  $\{0, 1\}^{n_b}$  consists of a set of keys  $\mathcal{K}_{\mathcal{F}}$  and a set  $\mathcal{F} = \{\text{prf}_k: \{0, 1\}^{n_a} \rightarrow \{0, 1\}^{n_b} \mid k \in \mathcal{K}_{\mathcal{F}}\}$ . When referring to the set  $\mathcal{F}$ , we will tacitly assume that the key space ( $\mathcal{K}_{\mathcal{F}}$ ) is known.

**SECURITY.** We follow Bellare et al. [25, 26] and formalise the PRF security of  $\mathcal{F}$  in Figure 2.1, where  $\text{Func}(\{0, 1\}^{n_a}, \{0, 1\}^{n_b})$  denotes the set of all functions with domain  $\{0, 1\}^{n_a}$  and range  $\{0, 1\}^{n_b}$ . The PRF advantage of an adversary  $\mathcal{A}$  against  $\mathcal{F}$  is defined as

$$\text{Adv}_{\mathcal{F}}^{\text{prf}}(\mathcal{A}) = 2 \cdot \Pr[\text{PRF}_{\mathcal{F}, \mathcal{A}}] - 1 .$$

This definition accounts for the fact that an adversary can always guess and succeed with probability  $1/2$  and normalises the advantage that an adversary's strategy provides (over the simple guessing strategy) to the interval  $[0, 1]$ . The expression  $\Pr[\text{PRF}_{\mathcal{F}, \mathcal{A}}]$  denotes the probability that the PRF game given in Figure 2.1 returns 1 (which signals that the adversary correctly guessed  $b$ ), when run with the adversary  $\mathcal{A}$  and set of functions  $\mathcal{F}$ ; this probability is taken over the random choices of  $b$ ,  $k$ ,  $\text{urf}$ , and any randomness used by the adversary.

### 2.2.2 Block ciphers

**BACKGROUND.** A block cipher (BC) is a symmetric encryption algorithm that operates on fixed-width blocks of plaintext and ciphertext; commonly, the blocks consist of 64 or 128 bits. There is no particular need to presume that the block cipher uses these common block sizes or that our plaintext and ciphertext are in bits, however we will assume the latter throughout.

## 2.2 Cryptographic background

---

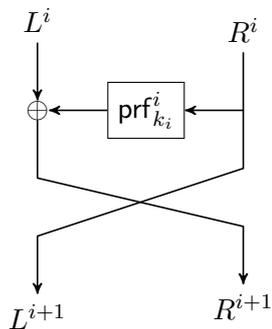


Figure 2.2: The round function of a Feistel cipher. The input to each round is split into two (possibly unevenly) and in the  $i^{\text{th}}$  round a function  $\text{prf}^i$  is used with round key  $k_i$  to compute the input to the next round.

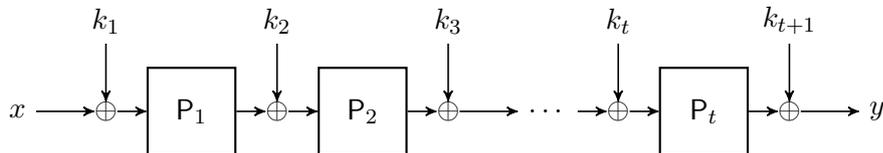


Figure 2.3: A  $t$ -round key-alternating cipher, with permutations  $P_1, \dots, P_t$  and round keys  $k_1, \dots, k_{t+1}$ .

Two popular design strategies for block ciphers are Feistel networks (or Luby–Rackoff ciphers) [45, 188] and key-alternating ciphers [91]. There are many other strategies such as Lai–Massey ciphers [172] and various generalisations of Feistel networks (e.g. [202, 136, 193]). These methods all result in iterated block ciphers, where a round function is repeatedly applied to process the plaintext and more rounds generally offers greater security. In a Feistel cipher, the plaintext is processed in two halves; the round function of a Feistel cipher consists of applying a keyed function to one half, xoring the output into the other half, and swapping the halves, as illustrated in Figure 2.2. In a key-alternating cipher the plaintext is processed by alternately xoring with a round key and applying a key-independent permutation, as shown in Figure 2.3. Commonly, the functions  $\text{prf}^1, \dots, \text{prf}^t$  or permutations  $P_1, \dots, P_t$  are closely related in order to simplify implementations. A more detailed discussion of key-alternating ciphers is given in Section 4.

In the 1970s, the National Bureau of Standards standardised the Data Encryption Standard (DES) based on Lucifer, a cipher developed by IBM [199]. Both Lucifer and DES have a Feistel structure, although the round functions are not PRFs. As it became clear that DES did not provide the level of security required, particularly

## 2.2 Cryptographic background

---

with respect to its key size, variants of DES were proposed as ways to increase the size of the keys. DES-X [161, 162] is one of the principal examples, where the extra key material is used as pre- and post-whitening keys and is xored with the inputs and outputs of DES; Triple DES [17, 32] is another example, where DES is applied to the plaintext three times using using independent keys (a two-key variant is also defined [17, p. 10]).

In 1997, the National Institute of Standards and Technology initiated the process of standardising a replacement for DES [212]. This involved 15 submissions, and after extensive analysis Rijndael was announced as the Advanced Encryption Standard (AES) [208]; AES is an example of a key-alternating cipher that is widely used, having been standardised for a range of applications as discussed in Chapter 1.

Since the standardisation of Rijndael, a popular theme in symmetric cryptography research has been the topic of lightweight cryptography. Here the motivation is to design and study primitives and schemes that are well suited to pervasive devices and are able to provide security despite scarce resources (e.g. small implementation footprint or low energy consumption). A number of lightweight block ciphers have been proposed, including HIGHT [137], Present [59], KATAN [67], LED [131], KLEIN [129], PRINCE [61], and Zorro [118].

From a block cipher, it is possible to build a wide range of primitives and schemes including symmetric encryption schemes (e.g. [23, 239]), message authentication codes (e.g. [199, Sect. 9.5.1] and [207]), authenticated encryption schemes (e.g. [272, 100]) and (universal) hash functions (e.g. [24]). Indeed, many commonly used symmetric algorithms are constructed from block ciphers—even if it is not immediately clear where they feature in the design, as is the case for the Secure Hash Algorithm (SHA) hash functions which are built from the SHA-CAL block cipher [134].

**SYNTAX.** A block cipher consists of a set of keys  $\mathcal{K}$  and a pair of functions  $E, D: \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$  such that for every  $k \in \mathcal{K}$  the map  $E(k, \cdot)$  is a permutation on  $\{0, 1\}^n$ . Such an  $E$  uniquely defines its inverse map  $D(k, \cdot)$  for each key  $k$ . We write  $BC = (E, D)$  to denote a block cipher, which also implicitly defines the block cipher's key space  $\mathcal{K}$  and the message space or domain  $\{0, 1\}^n$ . We will often denote  $E(k, \cdot)$  by  $E_k(\cdot)$  and the inverse of this permutation by  $D_k(\cdot)$ .

## 2.2 Cryptographic background

---

<u>SPRP<sub>BC, A</sub>:</u> $b \leftarrow \{0, 1\}; k \leftarrow \mathcal{K}$ $(iE, iD) \leftarrow \text{Block}(\mathcal{K}, \{0, 1\}^n)$ $b' \leftarrow \mathcal{A}^{\text{BC-ENC}, \text{BC-DEC}}$ Return $(b' = b)$	<u>BC-ENC(<math>X</math>):</u> If $b = 0$ Return $iE(k, X)$ Return $E(k, X)$  <u>BC-DEC(<math>X</math>):</u> If $b = 0$ Return $iD(k, X)$ Return $D(k, x)$
---	--

Figure 2.4: Game defining the SPRP security of a block cipher  $\text{BC} = (E, D)$ . In the PRP game the adversary cannot query the BC-DEC oracle.

SECURITY. We follow Luby and Rackoff [188] and formalise the strong pseudo-random permutation (SPRP) security of  $\text{BC}$  in Figure 2.4; we let  $\text{Block}(\mathcal{K}, \{0, 1\}^n)$  denote the set of all block ciphers with key space  $\mathcal{K}$  and domain  $\{0, 1\}^n$ . The SPRP advantage of an adversary  $\mathcal{A}$  against  $\text{BC}$  is defined as

$$\text{Adv}_{\text{BC}}^{\text{sprp}}(\mathcal{A}) = 2 \cdot \Pr[\text{SPRP}_{\text{BC}, \mathcal{A}}] - 1 .$$

The pseudorandom permutation (PRP) advantage is defined similarly, for adversaries that make no queries to BC-DEC. An adversary playing the SPRP game is said to be carrying out a chosen-ciphertext attack (CCA), while adversaries against the PRP security of a scheme are carrying out chosen-plaintext attacks (CPA).

There is an extensive literature showing that, with sufficiently many rounds, key-alternating and Feistel ciphers can meet this security requirement (e.g. [98, 73, 188]).

### 2.2.3 Cryptographic permutations

Public permutations have become a popular basis from which to build other symmetric primitives. In this setting, it is assumed that all parties have access to some specified permutations. Although several submissions to the SHA-3 competition have popularised this idea (including the eventual winner, Keccak [40, 210]), it can be dated back to at least Even and Mansour [106, 107]. A number of other hash function proposals explicitly use a public permutation, such as Quark [14], Spongent [58], and Photon [130]. Typically these instantiate the sponge framework [39] with a particular choice of public permutation.

Part of the motivation behind the recent interest in building primitives from pub-

## 2.2 Cryptographic background

---

lic permutations has been the flexibility offered by the sponge construction and the ability to have several primitives instantiated with schemes that share much of their implementation, which is particularly attractive for lightweight applications. However, public permutations occur (less transparently) in other schemes; for example, in a key-alternating cipher [91] the key-independent permutations are publicly known. In this thesis, a block cipher constructed from ideal permutations that are accessed via an oracle  $\pi$  is denoted  $\text{BC}^\pi = (\text{E}^\pi, \text{D}^\pi)$ .

One disadvantage of this paradigm is that, historically, it has not been clear how best to analyse the security of the scheme from a theoretical perspective. A common approach taken has been to analyse the scheme in the random-permutation model. In this model, the public permutation is replaced by a uniform, random permutation and it is assumed that an adversary only gains information about the value  $\text{P}(a)$  by explicitly evaluating  $\text{P}$  at the point  $a$  (beyond knowing that  $\text{P}(a_1) \neq \text{P}(a_2)$  for  $a_1 \neq a_2$ ). The random-permutation model is closely related to the random-oracle model [81], in which all parties have access to a public random function, as formalised by Bellare and Rogaway [31]. The ideal cipher model gives all parties access to a family of random permutations and dates back to Shannon [249, p. 691]. It has been used to reason about the security of many schemes (e.g. [56, 162]) and the random-permutation model simply gives access to an ideal cipher consisting of a single random permutation.

While analysis within these ideal models perhaps gives some indication of a scheme's security, it is possible to give constructions that are provably secure in these models but insecure regardless of how the ideal component is instantiated (e.g. [65, 194, 19, 51]). This has led to some discussion about the merits of such a model [164, 165].

### 2.2.4 Symmetric encryption schemes

BACKGROUND. Symmetric encryption schemes (SES) were perhaps the first cryptographic scheme to be used: the Caesar, Vigenère, and Playfair ciphers are examples of schemes intended to allow two parties, sharing a private key, to communicate with confidentiality; Kahn [155] gives a detailed discussion of historical encryption schemes. In 1917, Vernam [268] described what has become known as the one-time

## 2.2 Cryptographic background

---

pad, which Shannon [249] showed in 1948 to provide perfect secrecy. Even against adversaries with unlimited computational resources, symmetric encryption schemes providing perfect secrecy completely hide the plaintext; more formally, the probability distribution of plaintexts is identical to the distribution of plaintexts conditioned on the observed ciphertexts. The period between World War 1 and World War 2 saw a rapid development of mechanical, rotor-based enciphering machines, including Scherbius's Enigma [247].

Around the time that DES was standardised, attention turned to constructing symmetric encryption schemes from block ciphers. As block ciphers only describe how to encrypt fixed-width blocks, it is necessary to specify how to use a block cipher to encrypt longer or shorter messages. Additionally, block ciphers are deterministic whereas Goldwasser and Micali [127] describe the need for a secure encryption scheme to be probabilistic (so that sending the same plaintext twice is highly likely to result in different ciphertexts).

Modes of operation provide a solution to these two issues: notable examples include output feedback (OFB), cipher feedback (CFB), counter (CTR), and cipher block chaining (CBC) mode. In some cases (e.g. CBC mode) an appropriate padding scheme is needed in order to deal with partial final blocks. These modes of operation are analysed by Bellare et al. [23] in the concrete security framework.

Rogaway et al. [241] introduce the concept of nonce-based symmetric encryption, phrased in terms of authenticated encryption (AE) schemes; Rogaway [237, 239] formally describes nonce-based symmetric encryption and nonce-based schemes providing authenticated encryption with associated data (AEAD).

Previous work had recognised the need for probabilistic encryption and these works examine what level of security is achievable if the only assumption about the initialisation vector (IV) is that it is non-repeating (a nonce). As an example, with Counter mode it is sufficient to choose the IVs such that no counter value is repeated, whereas (textbook) CBC mode requires an unpredictable IV. This nonce-based method proposes that by clarifying and reducing the assumptions about the IV, schemes that are used in practice are less likely to be misused.

## 2.2 Cryptographic background

---

$\text{IND\$-CPA}_{\text{SES}, \mathcal{A}}:$ $b \leftarrow_{\$} \{0, 1\}; k \leftarrow_{\$} \mathcal{K}$ $b' \leftarrow_{\$} \mathcal{A}^{\text{SYM-ENC}}$ $\text{Return } (b' = b)$	$\text{SYM-ENC}(N, M):$ $\text{If } b = 0 \text{ Return } \$ (k, N, M)$ $\text{Return } \text{Enc}(k, N, M)$
---	--

Figure 2.5: Game defining the IND\$-CPA security of a symmetric encryption scheme  $\text{SES} = (\text{Enc}, \text{Dec})$ .

**SYNTAX.** Following Rogaway et al. [239], a nonce-based symmetric encryption scheme consists of two functions:  $\text{Enc}, \text{Dec}: \mathcal{K} \times \mathcal{N} \times \{0, 1\}^{\leq L_P} \rightarrow \{0, 1\}^{\leq L_C}$ , where  $L_P$  is a bound on the length of messages that may be encrypted using the scheme and  $L_C$  is a corresponding bound on the length of the ciphertext. Additionally, for every  $k, N$  and  $P$  we require that  $\text{Dec}(k, N, \text{Enc}(k, N, P)) = P$ . We denote a symmetric encryption scheme by  $\text{SES} = (\text{Enc}, \text{Dec})$ , which implicitly defines the sets  $\mathcal{K}$  and  $\mathcal{N}$  and the message-length bounds,  $L_P$  and  $L_C$ . The function  $\text{Enc}(k, \cdot, \cdot)$  will often be denoted  $\text{Enc}_k(\cdot, \cdot)$ , and similarly for  $\text{Dec}$ . We additionally assume that  $\text{Enc}$  is length-preserving, in that  $|\text{Enc}_k(N, P)| = |P|$  (excluding the transmission of the nonce); this is not the most general description that one could imagine, however all of the schemes considered in this thesis have this property.

**SECURITY.** The IND\$-CPA game (indistinguishability from random, under a chosen-plaintext attack) is formalised in Figure 2.5, where we assume that the length of the ciphertext,  $|\text{Enc}_k(N, M)|$ , depends only on  $|M|$  and define  $\$$  to be an oracle returning  $|\text{Enc}_k(N, M)|$  uniformly random bits. Throughout, we will only consider nonce-respecting adversaries, so that no adversary queries their oracle with pairs  $(N, M)$  and  $(N, M')$ . The adversary's IND\$-CPA advantage [241, 240] is defined as

$$\text{Adv}_{\text{SES}}^{\text{ind\$-cpa}}(\mathcal{A}) = 2 \cdot \Pr[\text{IND\$-CPA}_{\text{SES}, \mathcal{A}}] - 1 .$$

Initially, semantic security [127] was introduced as the objective of secure encryption (first for public-key schemes, then modified for symmetric schemes). This can be shown to be equivalent to ciphertext indistinguishability [128] and Bellare et al. [23] describe several indistinguishability notions, such as Find-then-Guess, Left-or-Right, and Real-or-Random; all are stronger (and easier to work with) than semantic security. IND\$ security trivially implies each of these notions, is easier to understand and motivate, and is often achieved by the schemes that we wish to reason about.

## 2.2 Cryptographic background

---

It is possible to strengthen each of these notions by permitting adversaries access to a decryption oracle (which returns the decryption of a ciphertext chosen by the adversary). In this thesis we will not consider chosen-ciphertext attacks against symmetric encryption schemes not described as authenticated encryption schemes and so the reader is referred to standard references (such as [157]) for further details.

### 2.2.5 Tweakable block ciphers

BACKGROUND. A tweakable block cipher (TBC) is a block cipher that admits an additional public input (the ‘*tweak*’) to introduce extra variability at the message-block level, in the same way that a nonce or IV introduces variability at the message level. Tweakable block ciphers are formalised by Liskov, Rivest, and Wagner [186].

The motivation for redrawing the abstraction boundary in this way is that, in many situations, independent instances of a block cipher are desirable. For example, in Electronic Codebook (ECB) mode, if a repeated ciphertext block is observed then an adversary can conclude that the corresponding plaintext blocks are identical; however if an independent instance of the block cipher were to be used for each block, then this would not be the case. One potential solution to this particular issue with ECB mode would be to re-key the block cipher with fresh, pseudorandom, nonce-dependent keys for each block processed. Most block ciphers incur a cost associated with changing the key, so a major motivation of Liskov, Rivest, and Wagner was to realise independent instances of a block cipher more efficiently than rekeying.

The Hasty Pudding Cipher [248] and Mercy [87] are early examples of ciphers that natively support a tweak (in the case of the Hasty Pudding Cipher, this was called Spice); Threefish [112] is a more recent example. It is also possible to build tweakable block ciphers from conventional block ciphers, as in Rogaway’s XE and XEX modes [238], or by modifying existing block cipher designs, as in Goldenberg et al.’s work on tweaking Luby–Rackoff ciphers [124]. Tweakable block ciphers can be used to construct other cryptographic functions, such as symmetric and authenticated encryption schemes (e.g. OCB [238, 170]), hash functions (e.g. Skein [112]), and message authentication codes (MACs, e.g. PMAC [55]).

## 2.2 Cryptographic background

---

$\text{TSPRP}_{\text{TBC}, \mathcal{A}}:$ $b \leftarrow_{\$} \{0, 1\}; k \leftarrow_{\$} \mathcal{K}$ $(i\text{TE}, i\text{TD}) \leftarrow_{\$} \text{Block}(\mathcal{K} \times \mathcal{T}, \{0, 1\}^n)$ $b' \leftarrow_{\$} \mathcal{A}^{\text{TBC-ENC}, \text{TBC-DEC}}$ $\text{Return } (b' = b)$	$\text{TBC-ENC}(T, X):$ $\text{If } b = 0 \text{ Return } i\text{TE}(k, T, X)$ $\text{Return } \text{TE}(k, T, X)$ $\text{TBC-DEC}(T, X):$ $\text{If } b = 0 \text{ Return } i\text{TD}(k, T, X)$ $\text{Return } \text{TD}(k, T, X)$
---	--

Figure 2.6: Game defining the TSPRP security of a tweakable block cipher  $\text{TBC} = (\text{TE}, \text{TD})$ . In the TPRP game the adversary cannot query the TBC-DEC oracle.

**SYNTAX.** A tweakable block cipher is a pair of functions  $\text{TE}, \text{TD}: \mathcal{K} \times \mathcal{T} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ , where  $\mathcal{K}$  is the key space,  $\mathcal{T}$  is the tweak space, and  $\{0, 1\}^n$  is the message space. We require that, for every  $k \in \mathcal{K}$  and  $T \in \mathcal{T}$ ,  $\text{TE}(k, T, \cdot)$  is a permutation on  $\{0, 1\}^n$  and we denote the inverse of this permutation by  $\text{TD}(k, T, \cdot)$ . We write  $\text{TBC} = (\text{TE}, \text{TD})$  to denote a tweakable block cipher, which also implicitly defines the key space  $\mathcal{K}$ , tweak space  $\mathcal{T}$ , and the message space  $\{0, 1\}^n$ . We will often denote  $\text{E}(k, T, \cdot)$  by  $\text{E}_k(T, \cdot)$  and the inverse of this permutation by  $\text{D}_k(T, \cdot)$ . The set of all tweakable block ciphers with key space  $\mathcal{K}$ , tweak space  $\mathcal{T}$ , and domain  $\{0, 1\}^n$  can be canonically mapped to  $\text{Block}(\mathcal{K} \times \mathcal{T}, \{0, 1\}^n)$ .

**SECURITY.** We use the definition of (strong) tweakable block cipher security given by Liskov et al. [186, 187]. Consider an adversary  $\mathcal{A}$  and define their tweakable-SPRP (TSPRP) advantage against  $\text{TBC}$  as

$$\mathbf{Adv}_{\text{TBC}}^{\text{tsprp}}(\mathcal{A}) = 2 \cdot \Pr[\text{TSPRP}_{\text{TBC}, \mathcal{A}}] - 1,$$

where the TSPRP game is as formalised in Figure 2.6.

The tweaked-PRP (TPRP) advantage is defined similarly, for adversaries that make no queries to TBC-DEC. Note that, in the definition of both tweakable block ciphers and strong tweakable block ciphers, the adversary is able to choose both the input and the tweak for each query.

## 2.2 Cryptographic background

---

### 2.2.6 Message authentication codes

BACKGROUND. Message authentication is a somewhat orthogonal aim to that of confidentiality: the objectives of message authentication are to derive some guarantees about the sender’s identity and the integrity of the message. Historically, message authentication was obtained by the decryption of a ciphertext to a ‘meaningful’ plaintext—this was perhaps in part due to the perceived unlikelihood of attackers injecting fake messages and the error propagation properties of the encryption schemes in use at the time.

Around the time that DES was standardised, the recognition that encryption does not provide message authentication and the importance of message authentication (particularly to the banking industry) led to the development of dedicated message authentication codes (MACs). Early examples of MACs include the Message Authentication Algorithm [92] and CBC-MAC, which is specified by the American National Standards Institute (ANSI) in ANSI X9.9 and X9.19 [199], and by the National Institute for Standards and Technology (NIST) in Federal Information Processing Standard (FIPS) 113 [207].

These two algorithms follow the (now) common pattern of gaining assurance about the origin of a message via a short ‘tag’ appended to the message. Informally this tag is a secure MAC if it is not feasible, without the key, to find a message and tag that appear to have been created by a legitimate sender. The first formal treatment of message authentication codes (and in particular CBC-MAC) dates from 1994 [25]; it is well known that any PRF is a good MAC [25, 126, 125].

There are four common types of message authentication code: those based on block ciphers, cryptographic hash functions, permutations, or universal hash functions. In the first category, CBC-MAC and its variants [54, 147] appear in many introductory courses on cryptography [192] and are widely used and standardised [142, 272]; XOR MACs are an alternative method to build a MAC from a block cipher [24]. Cryptographic hash functions are perhaps the most common building block for MACs; the hash-based message authentication code (HMAC) [167, 20] is a mode of operation for a hash function which realises a secure MAC and it is very widely

## 2.2 Cryptographic background

---

used [273, 158, 159, 209]. The documentation for Keccak (SHA-3) and sponge functions describe a wide variety of modes of operation for building primitives from a public permutation; one such mode results in a MAC [39, Sect. 3.2]. Universal hash functions have been studied extensively and an early suggested application was that of information-theoretic message authentication; see Section 2.2.7 for more details. It is worth remarking that this informal classification does not uniquely map MAC schemes to a category: for example, XOR MACs are fundamentally a block-cipher-based universal hash function that is then transformed into a MAC [24] and many hash-function-based MACs are instantiated with a hash function constructed from a block cipher.

**SYNTAX.** We will follow Black et al. [53] for a description of the syntax of nonce-based message authentication schemes. Message authentication schemes are not typically nonce-based, as it is straightforward to construct a nonce-based MAC from a MAC that does not require nonces by simply including the nonce with the message (and ensuring that the nonce can be distinguished from the message). However, nonces are required for some universal-hash-based schemes; in particular, the schemes that we study in Chapter 3 are all nonce-based.

A nonce-based message authentication scheme is a function  $\text{MAC}: \mathcal{K} \times \mathcal{N} \times \{0, 1\}^* \rightarrow \{0, 1\}^{|\tau|}$ , where  $\mathcal{K}$  is the key space,  $\mathcal{N}$  the nonce space,  $\{0, 1\}^*$  the message space, and  $\{0, 1\}^{|\tau|}$  the set of possible authentication tags. The authenticity of a tuple  $(N, M, \tau)$  is verified by computing  $\text{MAC}(k, N, M)$ : if  $\tau = \text{MAC}(k, N, M)$  then the tag is valid, otherwise it is invalid.

**SECURITY.** An adversary attacking a message authentication scheme  $\text{MAC}$  is given access to a tagging oracle  $\text{MAC-TAG}$  and a verification oracle  $\text{MAC-VER}$ . At the beginning of the experiment a key  $k$  is chosen uniformly at random, then  $\text{MAC-TAG}$  takes queries  $(N, M)$  and returns  $\text{MAC}(k, N, M)$ . The verification oracle takes queries  $(N, M, \tau)$  and returns 1 if  $\tau = \text{MAC}(k, N, M)$  or 0 otherwise. An adversary is said to successfully forge an authentication tag if they can produce a verification query  $(N, M, \tau)$  so that  $\text{MAC-VER}$  returns 1 when  $(N, M)$  was not previously queried to  $\text{MAC-TAG}$ .

## 2.2 Cryptographic background

---

$\text{FORGE}_{\text{MAC}, \mathcal{A}}:$ $k \leftarrow \mathcal{K}$ $(N, M, \tau) \leftarrow \mathcal{A}^{\text{MAC-TAG}, \text{MAC-VER}}$ <p style="margin-left: 20px;">If <math>(N, M)</math> not queried to MAC-TAG:</p> $\text{Return MAC-VER}(N, M, \tau)$ <p style="margin-left: 20px;">Else Return 0</p>	$\text{MAC-TAG}(N, M):$ $\text{Return MAC}(k, N, M)$ $\text{MAC-VER}(N, M, \tau):$ $\text{Return } (\text{MAC}(k, N, M) = \tau)$
--	---

Figure 2.7: Game defining the unforgeability of a message authentication scheme MAC.

We define the adversary’s advantage as

$$\text{Adv}_{\text{MAC}, \mathcal{A}}^{\text{forge}} = \Pr[\text{FORGE}_{\text{MAC}, \mathcal{A}}],$$

where the game FORGE is defined in Figure 2.7.

A common restriction of this security notion is to nonce-respecting adversaries where, although the adversary can control the nonce, they never query MAC-TAG for  $(N, M')$  if they have previously queried MAC-TAG for  $(N, M)$ .

One can distinguish between strong unforgeability and weak unforgeability [7]; typically this distinction is not made in the nonce-based setting that we focus on throughout this thesis. The notion we describe is analogous to the strong unforgeability game: an adversary wins if they can create a pair  $(M^*, \tau^*) \neq (M, \tau)$  for any  $(M, \tau)$  queried to MAC-TAG; weak unforgeability additionally requires that  $M^* \neq M$ .

### 2.2.7 Universal hash functions

BACKGROUND. In contrast to the common formulation of a cryptographic hash function as a single, publicly known function, a universal hash function family contains a number of different hash functions. Carter and Wegman [69, 70] introduce universal hash function families; their motivating problem is hashing for storage and retrieval on keys with an important performance consideration to minimise the number of collisions.

Prior to the definition of universal hash function families, Gilbert, MacWilliams, and Sloan [119] initiated the study of information-theoretic message authentication (from

## 2.2 Cryptographic background

---

a coding theory perspective); they credit Simmons with identifying the problem. Here, the objective is to convince a receiver that a particular message was indeed sent by the sender, rather than a (possibly) computationally-unlimited adversary; this is the analogue of an encryption scheme providing perfect secrecy [249].

In their 1979 paper, Wegman and Carter [270] apply universal hash function families to information-theoretic message authentication. They introduce strongly universal hash function families (of which, they observe, polynomials with bounded degree are an example) and describe how a member of such a family can be used to authenticate a single message. Additionally, multiple messages can be authenticated if the output of the universal hash function is encrypted with a one-time pad; Brassard [63] observes that using a stream cipher as a pseudo-one-time pad reduces the security of an unconditional authenticator to the (computational) security of the stream cipher, whilst dramatically reducing the amount of key material used.

Simmons and Stinson have both made significant contributions to this area: Simmons [253] gives a survey on the general theory of unconditional authentication, which includes an overview of several of his papers (such as [254, 255, 256]); and Stinson has comprehensively studied the combinatorics of codes providing unconditional authentication [259, 260, 261, 262, 262, 263].

Krawczyk [168] introduces the notion of a family of hash functions being  $\varepsilon$ -otp-secure, which is to say that an adversary succeeds with probability at most  $\varepsilon$  when attempting to forge an authentication tag that is constructed using the one-time pad encryption of hash value of the message. A necessary and sufficient condition for a hash function family to be  $\varepsilon$ -otp-secure is identified, which is now commonly known as  $\varepsilon$ -almost XOR universalilty ( $\varepsilon$ -AXU), following Rogaway [235, 236]. A more general condition,  $\varepsilon$ -almost  $\Delta$  universal was introduced by Stinson; this recognises the fact that any commutative group operation ( $\Delta$ ) could be used in place of xor. In this thesis we will generally refer to  $\varepsilon$ -AXU hash function families; however any remark made that requires an  $\varepsilon$ -AXU hash function family in characteristic 2 will also hold for an  $\varepsilon$ -almost strongly universal [263] or  $\varepsilon$ -almost  $\Delta$  universal [262] hash function family (in a finite field with group operation  $\Delta$ ).

Krawczyk [168] also gives two constructions of  $\varepsilon$ -almost XOR universal hash function

## 2.2 Cryptographic background

---

families; both of these constructions require relatively short keys to specify a member of the hash function family. The first is a cryptographic cyclic redundancy code (CRC) which closely resembles Rabin’s fingerprinting codes [228]; the second constructs a Toeplitz matrix (in which each diagonal is constant) from a linear feedback shift register, building on proposals from Carter and Wegman [70] and Mansour, Nisan, and Tiwari [191]. Shoup [251] describes several methods for realising universal hash function families that are related to polynomials. These proposals include the evaluation hash [42, 94, 264] and the division hash or cryptographic CRC of Krawczyk [168], plus a generalised form that includes both as a special case; the main example of interest to this thesis is the evaluation hash and a more thorough discussion of this construction is given in Chapter 3.

More recently, several universal hash function families have been proposed, including Rogaway’s Bucket Hash [235, 236], XOR MAC [24] and some constructions that follow the design principles of MMH [132] (e.g. [53, 57]).

**SYNTAX.** A family of hash functions will be denoted

$$\mathcal{H} = \{h_H : \{0, 1\}^* \rightarrow \{0, 1\}^n \mid H \in \mathcal{K}_{\mathcal{H}}\} ,$$

with each hash function  $h_H$  indexed by a key  $H \in \mathcal{K}_{\mathcal{H}}$ .

**SECURITY.** A family of hash functions  $\mathcal{H} = \{h_H : \{0, 1\}^* \rightarrow \{0, 1\}^n \mid H \in \mathcal{K}_{\mathcal{H}}\}$  is said to be a  $\text{universal}_2$  hash function family [69, 70] if for every  $M, M' \in \{0, 1\}^*$  with  $M \neq M'$

$$\Pr_{H \in \mathcal{K}_{\mathcal{H}}} [h_H(M) = h_H(M')] \leq \frac{1}{2^n} .$$

A family of hash functions  $\mathcal{H} = \{h_H : \{0, 1\}^* \rightarrow \{0, 1\}^n \mid H \in \mathcal{K}_{\mathcal{H}}\}$  is said to be a  $\text{strongly universal}_s$  hash function family [270, 271] if for every sequence  $M_1, \dots, M_s$  of distinct elements of  $\{0, 1\}^*$  and for every sequence  $y_1, \dots, y_s$  of elements of  $\{0, 1\}^n$ ,

$$\Pr_{H \in \mathcal{K}_{\mathcal{H}}} [\forall i, h_H(M_i) = y_i] \leq \frac{1}{2^{ns}} .$$

These two definitions can be slightly relaxed: if the relevant probability is bounded by  $\varepsilon$  instead of by  $\frac{1}{2^n}$  or  $\frac{1}{2^{ns}}$ , then the hash function family are said to be  $\varepsilon$ -almost universal or  $\varepsilon$ -almost strongly universal respectively.

## 2.2 Cryptographic background

---

A family of hash functions is said to be  $\varepsilon$ -almost XOR universal [168, 235] if for every  $M, M' \in \{0, 1\}^*$  with  $M \neq M'$  and for every  $c \in \{0, 1\}^n$ ,

$$\Pr_{H \in \mathcal{K}_H} [\mathbf{h}_H(M) \oplus \mathbf{h}_H(M') = c] \leq \varepsilon .$$

We abbreviate  $\varepsilon$ -almost XOR universal to  $\varepsilon$ -AXU and generically refer to hash function families satisfying conditions similar to the above as universal hash functions.

### 2.2.8 Authenticated encryption

BACKGROUND. As noted earlier, the aims of confidentiality and authenticity are distinct. The one-time pad provides perfect secrecy, but offers absolutely no authenticity: an adversary can flip specific bits in the plaintext by flipping the same bits in the ciphertext. Similarly, universal-hash-based MACs can provide information-theoretically secure authentication, as can a universal-hash-based MAC concatenated with the plaintext, which trivially offers no confidentiality.

It has become increasingly clear that the security properties users expect from symmetric cryptography are better described in terms of authenticated encryption (AE) schemes [29], where the plaintext is both encrypted and authenticated. Authenticated encryption with associated data (AEAD) schemes [237] additionally protect data that should be authenticated but not encrypted, such as addresses or routing information. For the vast majority of applications this provides the required functionality and reduces the risk that the intended security is compromised when schemes providing confidentiality and authenticity are combined. This choice of abstraction boundary has become a common interface in cryptographic libraries, with TLS 1.3 currently planned to only support AEAD ciphers [230, Sect. 6.2.2] and the NaCl ‘crypto\_box’ construction realising this model [38].

Bellare and Namprempre [29] formally analyse the idea of generic composition, where the security of the combination depends only on the security properties of the underlying schemes; a similar approach is adopted by Krawczyk [169] and Krawczyk and Cannetti [66]. Part of the motivation for these works was that three major internet security protocols all constructed AEAD schemes in a different way: Secure Shell (SSH) encrypted and authenticated the plaintext independently [273],

## 2.2 Cryptographic background

---

Internet Protocol Security (IPsec) first encrypted the plaintext then authenticated the resulting ciphertext [160], and Secure Socket Layer/Transport Layer Security (SSL/TLS) authenticated the plaintext then encrypted both the plaintext and the MAC [116]. Bellare and Namprempre’s analysis supports the approach of IPsec; several attacks have arisen from the poor choices made when constructing AEAD schemes in SSL/TLS and SSH (e.g. [267, 6]). This question has recently received renewed attention from Namprempre, Rogaway, and Shrimpton [205], where they consider the same question in the nonce-based setting and describe a more diverse set of secure constructions.

These results on generic composition enable the construction of a secure AEAD mode from any IND-CPA secure encryption scheme and any message authentication scheme that guarantees the integrity of ciphertexts (INT-CTXT) [29]. However dedicated constructions may offer advantages such as greater efficiency (with respect to speed, implementation area, or energy consumption) or alternative security properties (such as nonce-misuse or side-channel-attack resilience).

Perhaps the most widely used dedicated AEAD scheme is Galois/Counter Mode (GCM), which was specified by McGrew and Viega [197]; more details on the specification of GCM are given in Chapter 3. In addition to GCM, several other dedicated AE and AEAD schemes have been proposed such as OCB [238, 170] and several submissions to the eSTREAM competition [233]. In 2013, the CAESAR competition was launched [79], supported by NIST, aiming to *‘identify a portfolio of authenticated ciphers that (1) offer advantages over AES-GCM and (2) are suitable for widespread adoption’*; it has received over 50 submissions.

**SYNTAX.** Following Rogaway [237], an authenticated encryption with associated data (AEAD) scheme is an authenticated encryption (AE) scheme [29] that also authenticates some unencrypted data (the associated data). More formally, an AEAD scheme consists of two functions:  $\text{Enc}: \mathcal{K} \times \mathcal{N} \times \{0, 1\}^{\leq L_A} \times \{0, 1\}^{\leq L_P} \rightarrow \{0, 1\}^{\leq L_C}$  and  $\text{Dec}: \mathcal{K} \times \mathcal{N} \times \{0, 1\}^{\leq L_A} \times \{0, 1\}^{\leq L_C} \rightarrow \{0, 1\}^{\leq L_P} \cup \{\perp\}$ . As before  $L_A$ ,  $L_P$ , and  $L_C$  are bounds on the length of associated data, plaintext, and ciphertext (respectively) that can be processed by the scheme. We require that for every  $k \in \mathcal{K}$ ,  $N \in \mathcal{N}$ ,  $A \in \{0, 1\}^{\leq L_A}$ , and  $P \in \{0, 1\}^{\leq L_P}$  we have that  $\text{Dec}(k, N, A, \text{Enc}(k, N, A, P)) = P$ .

## 2.2 Cryptographic background

---

$\text{AEAD}_{\text{AEAD}, \mathcal{A}}:$ $b \leftarrow_{\$} \{0, 1\}; k \leftarrow_{\$} \mathcal{K}$ $b' \leftarrow_{\$} \mathcal{A}^{\text{AE-ENC}, \text{AE-DEC}}$ $\text{Return } (b' = b)$	$\text{AE-ENC}(N, A, P):$ $\text{If } b = 0 \text{ Return } \$(k, N, A, P)$ $\text{Return Enc}(k, N, A, P)$ $\text{AE-DEC}(N, A, C):$ $\text{If } b = 0 \text{ Return } \perp$ $\text{Return Dec}(k, N, A, C)$
---	---

Figure 2.8: Game defining the AEAD security of an AEAD scheme  $\text{AEAD} = (\text{Enc}, \text{Dec})$ .

Additionally, we assume that  $\text{Enc}$  is length-preserving, in that  $|\text{E}_k(N, A, P)| = |P| + |\tau|$  where  $|\tau|$  is fixed by the scheme; this is not the most general description imaginable, however all of the schemes considered in this thesis have this property.

**SECURITY.** There are two aspects to the security objective for an AEAD scheme: it should provide privacy of the plaintexts and integrity of the whole message (nonce, associated data and ciphertext).

The advantage of an adversary against an AEAD scheme will be measured by

$$\mathbf{Adv}_{\text{AEAD}}^{\text{aead}}(A) = 2 \cdot \Pr[\text{AEAD}_{\text{AEAD}, \mathcal{A}}] - 1,$$

where the AEAD game is as formalised in Figure 2.8, where as before,  $\$$  is an oracle that outputs  $|\text{E}(k, N, A, P)|$ -many random bits. This all-in-one notion of AEAD security is similar to that described by Rogaway and Shrimpton [242].

This definition is a slight strengthening of earlier definitions, as we insist that the scheme meets the definition of IND $\$$ -CPA security (described in Section 2.2.2); Bellare and Namprempre consider generic composition with IND-CPA secure encryption schemes [29]. A common restriction of this security definition is to only consider nonce-respecting adversaries, so no adversary queries the AE-ENC oracle with pairs  $(N, M)$  and  $(N, M')$ .

### 2.3 Security models

The indistinguishability and unforgeability notions described above have been widely adopted within the community as representations of the expected security properties for various primitives and schemes. However, there are applications and use cases for which it is desirable (or necessary) for the security model to be expanded in order to encompass other patterns of adversarial behaviour. In this section, we will describe some of the alternative security models that are used throughout this thesis.

#### 2.3.1 Weak keys

BACKGROUND. For any cryptographic algorithm, a relevant question for its security assessment is whether it contains weak keys. Informally, a weak-key class is a set of keys that cause some undesirable behaviour in the algorithm when they are used. The overall security implications of weak keys on an algorithm depends on the number of weak keys, the ease with which their use can be detected, and how undesirable the algorithm's behaviour is when a weak key is used.

A common design goal for block and stream ciphers is that the algorithm has no weak keys (a 'flat' key space); there are several examples of ciphers that do not meet this requirement, such as IDEA [89], Blowfish [266], RC4 [113] and DES [206, 201]. The most well known weak-key class is arguably that of DES; this class consists of keys  $k$  such that  $E_k(E_k(M)) = M$ . In this case the membership test is very simple however the weak-key class consists of only four keys.

In general, small weak-key classes do not significantly detract from an algorithm's security, however in some modes (such as if the algorithm is used to construct a hash function in the Davies-Meyer construction [221]) the adversary has control over which key is used and could cause a weak key to be used repeatedly.

DEFINITIONS. The Encyclopedia of Cryptography and Security [46] gives the following definition:

## 2.3 Security models

---

The strength of an encryption function  $E_k(P)$  may differ significantly for different keys  $k$ . If for some set  $WK$  of keys, the encryption function is much weaker than for others, this set is called a class of weak keys. The attack technique that succeeds against the keys in the class  $WK$  is called a membership test for the class.

Throughout this thesis, we use Handschuh and Preneel’s definition [135, Sect. 3.1]:

In symmetric cryptology, a class of keys  $\mathcal{D}$  is called a weak key class if for the members of that class the algorithm behaves in an unexpected way and if it is easy to detect whether a particular unknown key belongs to this class. For a MAC algorithm, the unexpected behavior can be that the forgery probability for this key is substantially larger than average. Moreover, if a weak key class  $\mathcal{D}$  is of size  $C$ , one requires that identifying that a key belongs to this class requires testing fewer than  $C$  keys by exhaustive search and fewer than  $C$  verification queries.

### 2.3.2 Related-key attacks

BACKGROUND. Formal analyses of cryptographic protocols often assume that cryptosystems are run on keys that are independently generated and bear no relation to each other. Implicit in this assumption is the premise that user keys are stored in protected areas that are hard to tamper with. Security under related-key attacks (RKAs), first identified by Biham and Knudsen [44, 43, 163], considers a setting where an adversary might be able to disturb user keys (perhaps by injecting faults [8]), and consequently run a cryptosystem on related keys. Resilience against RKAs has become a desirable security goal, particularly for block ciphers.

The need for RKA security is further highlighted by the fact that through (improper) design, a higher-level protocol might run a lower-level one on related keys. Prominent examples are the key derivation procedures in standardised protocols such as EMV [102] and the 3GPP integrity and confidentiality algorithms [146], where efficiency considerations have led the designers to use a block cipher under

## 2.3 Security models

---

related keys. Similar considerations can arise in the construction of tweakable block ciphers, if a block cipher is called on keys that are offset by xoring tweak values [186]. An RKA-secure primitive can offer security safeguards against such protocol misuse.

Bellare and Kohno [27] initiate the theoretical treatment of security under related-key attacks and propose definitions for RKA-secure pseudorandom functions and pseudorandom permutations. This model was subsequently extended by Albrecht et al. [5] to idealised models of computation, accounting for the possibility that keys may be derived in ways that depend on the ideal primitive. Both works prove that the ideal cipher is RKA secure against wide sets of related-key deriving (RKD) functions. Bellare and Cash [21] present an RKA-secure pseudorandom function from standard intractability assumptions and Bellare, Cash, and Miller [22] give a comprehensive treatment of RKA security for various cryptographic primitives, leveraging the RKA resilience of PRGs to construct RKA-secure instances of several other primitives. In Chapter 4 we consider the RKA security of block ciphers.

**RKD FUNCTIONS.** A related-key deriving (RKD) function maps keys to keys in some key space  $\mathcal{K}$ ; we view RKD functions as circuits that may contain special oracles gates  $\pi$ . An RKD set  $\Phi$  is a set of RKD functions  $\phi^\pi : \mathcal{K} \rightarrow \mathcal{K}$ , where  $\pi$  is an oracle; we assume that membership in RKD sets can be efficiently decided.

**RKA SECURITY.** Following Bellare and Kohno [27] and Albrecht et al. [5], we formalise the RKA security of a block cipher  $\text{BC}^\pi = (\text{E}^\pi, \text{D}^\pi)$  in the (multiple) ideal-permutation model via the game shown in Figure 2.9. Note that this game includes a procedure for oracle  $\pi$  defined above.

Access to multiple ideal permutations is equivalent to providing access to a block cipher  $\text{P}$  with key space  $[t]$ , where  $\text{P}_i(x) = \text{P}(i, x)$ . In order to ease notation, we define a single oracle  $\pi$ , which provides access to all  $t$  ideal permutations in both directions. This oracle takes as input  $(i, x, \sigma)$ , where  $i \in [t]$ ,  $x \in \{0, 1\}^n$ , and  $\sigma \in \{+, -\}$  and returns  $\text{P}_i(x)$  if  $\sigma = +$  and  $\text{P}_i^{-1}(x)$  if  $\sigma = -$ . Slightly abusing notation, we define  $\text{P}_i^\sigma(x) = \text{P}^\sigma(i, x) = \pi(i, x, \sigma)$ , and assume  $\sigma = +$  whenever it is omitted from the superscript.

Bellare and Kohno [27] show that it is not possible for any scheme to provide security

## 2.3 Security models

$\text{RKCCA}_{\text{BC}^\pi, \mathcal{A}, \Phi, t}:$ $b \leftarrow_{\$} \{0, 1\}; k \leftarrow_{\$} \mathcal{K}$ $(P, P^{-1}) \leftarrow_{\$} \text{Block}([t], \{0, 1\}^n)$ $(iE, iD) \leftarrow_{\$} \text{Block}(\mathcal{K}, \{0, 1\}^n)$ $b' \leftarrow_{\$} \mathcal{A}^{\text{RK-ENC, RK-DEC}, \pi}$ Return $(b' = b)$	$\text{RK-ENC}(\phi^\pi, x):$ $k' \leftarrow \phi^\pi(k)$ If $b = 0$ Return $iE(k', x)$ Return $E^\pi(k', x)$
$\pi(i, x, \sigma):$ Return $P^\sigma(i, x)$	$\text{RK-DEC}(\phi^\pi, x):$ $k' \leftarrow \phi^\pi(k)$ If $b = 0$ Return $iD(k', x)$ Return $D^\pi(k', x)$

Figure 2.9: Game defining the  $\Phi$ -RKCCA security of a block cipher  $\text{BC}^\pi = (E^\pi, D^\pi)$  with access to  $t$  ideal permutations. An adversary can query the RK-ENC and RK-DEC oracles with a  $\phi^\pi \in \Phi$  only. In the RKCPA game the adversary cannot query the RK-DEC oracle.

against an adversary that can request encryptions under arbitrary functions of the key. As a result of this, the RKA game is parametrised by an RKD set  $\Phi$  which specifies the RKD functions that an adversary is permitted to query during its attack. We define the advantage of an adversary  $\mathcal{A}$  carrying out a related-key chosen-ciphertext attack (RKCCA) against a block cipher  $\text{BC}$  via

$$\text{Adv}_{\text{BC}^\pi, \Phi, t}^{\text{rkcca}}(\mathcal{A}) = 2 \cdot \Pr [\text{RKCCA}_{\text{BC}^\pi, \mathcal{A}, \Phi, t}] - 1 .$$

The game and advantage for related-key chosen-plaintext attacks (RKCPA) are defined similarly by considering adversaries that do not make any RK-DEC queries (backwards queries to the permutations are still permitted). Denoting these games by RKCPA and RKCCA is a slight abuse of notation: we only consider these attacks against the PRP or SPRP security of a block cipher, whereas one could also consider (for example) related-key attacks against the TPRP or TSPRP security of a tweakable block cipher.

### 2.3.3 Indifferentiability

BACKGROUND. Maurer, Renner, and Holenstein [194] introduce a framework which formalises what it means for a non-monolithic object to be able to replace a (possibly idealised) component in arbitrary cryptosystems. This framework, known as indifferentiability, has been used to validate the design principle behind many cryp-

## 2.3 Security models

---

tographic constructions (e.g. [41, 10, 9]).

The motivation for this notion is to better model the situation in which components of a system depend on each other and an adversary may learn partial information about the randomness or state of these components. As an example, AES can be considered to call the AES round functions as subroutines and, as these functions are publicly known, it is possible for an adversary to evaluate these functions at points of their choosing and hence learn some information about them.

Similar examples can be taken from iterated hash functions: the Merkle-Damgård construction [80] calls a compression function; sponges [41] call the sponge permutation; and block-cipher-based compression functions may call a block cipher with a known key (see [222] for a thorough exposition of possible constructions).

Similarly to the random-permutation model, as discussed in Section 2.2.3, the correct interpretation of an indistinguishability proof is not clear. Ristenpart, Shacham, and Shrimpton [231] demonstrate that although indistinguishability implies indistinguishability, it does not necessarily guarantee composition in multi-stage settings. They emphasise that indistinguishability is still a worthwhile design objective (there are many practically important security notions covered by single-stage games), but it does not necessarily exclude ‘structure-abusing attacks’, as had been suggested.

Informally, a multi-stage setting is one in which the adversary can be decomposed into a number of ‘smaller’ algorithms passing limited state between each other; this should be compared with the single, stateful adversary present in the indistinguishability and unforgeability notions described above. Ristenpart et al. go on to explain that multi-staged security notions includes deterministic or searchable public-key encryption, as well as security in the presence of key-dependent messages. Related-key security is also a two-stage game if the RKD functions can depend on public oracles; in this case the RKD functions take the role of the other adversarial stage.

**SECURITY MODEL.** The approach taken by Maurer et al. [194] considers the general situation of a system with many interacting components. In this thesis it is sufficient to consider just two components: a ‘construction’ built from a ‘primitive’.

### 2.3 Security models

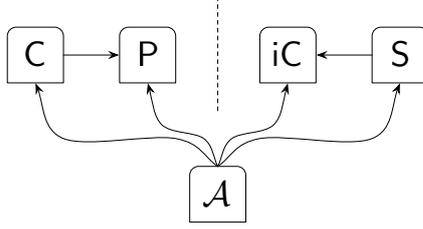


Figure 2.10: The indistinguishability framework. An adversary  $\mathcal{A}$  interacts with either: an ideal primitive  $\mathsf{P}$  and the real construction  $\mathsf{C}$ , which can call the ideal primitive; or an ideal construction  $\mathsf{iC}$  and a simulator  $\mathsf{S}$ , which can query the ideal construction.

$\text{INDIFF}_{\mathsf{C},\mathcal{A}}:$ $b \leftarrow_{\$} \{0, 1\}; k \leftarrow_{\$} \mathcal{K}$ $\mathsf{iC} \leftarrow_{\$} \mathcal{I}$ $b' \leftarrow_{\$} \mathcal{A}^{\text{CONST},\text{PRIM}}$ $\text{Return } (b' = b)$	$\text{CONST}(X):$ $\text{If } b = 0 \text{ Return } \mathsf{iC}(k, X)$ $\text{Return } \mathsf{C}^{\mathsf{P}}(k, X)$ $\text{PRIM}(X):$ $\text{If } b = 0 \text{ Return } \mathsf{S}^{\mathsf{iC}}(X)$ $\text{Return } \mathsf{P}(X)$
---	---

Figure 2.11: Game defining the indistinguishability of a construction  $\mathsf{C}$ . The idealised primitive is denoted by  $\mathsf{P}$ , the idealised construction by  $\mathsf{iC}$  and the simulator by  $\mathsf{S}$ . The set  $\mathcal{I}$  denotes all possible idealised constructions with the required syntax.

The indistinguishability framework permits an adversary to interact with either the real construction and an idealised primitive, or an idealised scheme and a simulated primitive. That is, in the case that an adversary is interacting with the idealised scheme (in place of the construction), a simulator responds to the adversary’s queries with ‘how the primitive should respond, given how the ideal construction is behaving’. The fundamental difference between these two settings is that in one case the construction calls the primitive as a subroutine, while in the other the simulator (playing the role of the primitive) calls the idealised construction.

We define the indistinguishability advantage of an adversary  $\mathcal{A}$  via

$$\text{Adv}_{\mathsf{C}}^{\text{indiff}}(\mathcal{A}) = 2 \cdot \Pr[\text{INDIFF}_{\mathsf{C},\mathcal{A}}] - 1 .$$

## 2.3 Security models

---

### 2.3.4 Algebraic cryptanalysis

The ideas underpinning algebraic cryptanalysis date back at least as far as Shannon [249], who describes how an adversary could perhaps solve a system of equations in order to recover the key. Algebraic attacks are powerful cryptanalytic techniques that have been used to attack a wide range of schemes: several stream ciphers have been broken (e.g. [84, 82]), while the technique has also been applied (generally unsuccessfully) to block ciphers [76, 77]. Perhaps the best known algebraic attacks are those against AES [85], although the practicality of using the XSL algorithm to attack AES has since been questioned [75].

Algebraic cryptanalysis consists of two relatively distinct phases. The first is to write the entire cryptographic operation as a large set of multivariate polynomial equations, relating unknown values (such as parts of the key, intermediate values, or a target plaintext) to known, observed values (such as parts of plaintext/ciphertext pairs). The second phase is to solve this system of equations; common techniques include Gröbner Basis algorithms [110], linearisation [83], mixed-integer linear programming [62], and Boolean satisfiability (SAT) solvers [16]. A major strength of this cryptanalytic technique is that it is not statistical (as linear and differential cryptanalysis are) and often very few plaintext/ciphertext pairs are required. Unfortunately, the running time of algebraic attacks is often not well understood; the performance of these algorithms heavily depends on how the cryptosystem is represented and the particular structure of the derived equations [204, 76].

In Chapter 6 we will use Gröbner Basis algorithms as part of an algebraic attack; we refer the reader to standard texts (such as [86]) for further details on the implementation of these algorithms.

# Polynomial-based hash functions

---

## Contents

<b>3.1</b>	<b>Introduction</b>	<b>43</b>
<b>3.2</b>	<b>Prior work</b>	<b>44</b>
<b>3.3</b>	<b>Algebraic structure and its implications</b>	<b>52</b>
<b>3.4</b>	<b>Algebraic structure of previous attacks</b>	<b>60</b>
<b>3.5</b>	<b>Weak keys</b>	<b>63</b>
<b>3.6</b>	<b>GCM with short multiplications</b>	<b>66</b>
<b>3.7</b>	<b>Square Hash</b>	<b>70</b>
<b>3.8</b>	<b>Discussion</b>	<b>73</b>

---

*In this chapter, we study universal hash functions that are constructed via the evaluation of a polynomial in a finite field. These are commonly used in schemes for message authentication and are popular due to their information-theoretic security guarantees, as well as their speed and simplicity. We describe a forgery attack that generalises all existing attacks on these schemes and identify a large number of weak keys. The particular relevance of these observations to GCM/2<sup>+</sup> is discussed and finally we describe many, large weak-key classes for MACs based on Square Hash, another universal hash function.*

*The work described in this chapter is joint work with Carlos Cid, appearing as [226] and [227]; it is available at [225].*

## 3.1 Introduction

Universal hash functions can be used to build message authentication codes and related constructions feature as components in AE and AEAD schemes. Commonly, the universal hash functions are based on polynomial evaluation: such schemes are widely standardised (e.g. [99, 166, 138, 181, 245]). McGrew and Viega’s Galois/Counter Mode (GCM) [197] is the most widely deployed example, Bernstein’s Poly1305 [34] is also widely used.

This approach is well studied and generally believed to be secure, as described in Section 2.2.7: Wegman and Carter [270] first proposed the use of universal hash functions (combined with the one-time pad) for message authentication in 1979; Brassard [63] describes using a stream cipher in place of the one-time pad, making the construction more efficient for multiple messages; and Krawczyk [168] describes a necessary and sufficient condition for a universal hash function family to provide security when used in this setting, giving several examples of constructions that meet this condition.

Previous works have focused on GCM due to its popularity, although their results often apply equally to many hash functions based on polynomial evaluation. There are a small number of papers attacking GCM via the universal hash component: Ferguson’s attack against truncated tags [111] demonstrates that the security of short tags is significantly lower than would be expected; Joux’s ‘forbidden attack’ [152] illustrates GCM’s brittleness under nonce reuse; Handschuh and Preneel describe methods to extend Joux’s attack [135]; and Saarinen’s cycling attacks [244] highlight a weakness due to the underlying algebraic structure of the hash function. Both Handschuh and Preneel [135] and Saarinen [244] describe classes of weak keys for polynomial-evaluation-based universal hash functions.

The main motivation for this work was the observation that these attacks against GCM are algebraic in nature, and seem to exploit a fundamental underlying algebraic structure of the polynomial-based hash function which we discuss in this chapter.

## 3.2 Prior work

---

### Contributions

The contributions of this chapter are to identify and study some of the properties of hash functions based on polynomial evaluation that arise from this underlying algebraic structure. As a result, we are able to describe a general forgery attack against related message authentication schemes which removes several limitations of previous works: our attack can be used with short messages, does not require nonce-reuse or truncated tags, applies regardless of the field in which the hash is evaluated, and additionally facilitates length extension attacks. Furthermore, we provide a common description of all published attacks against these schemes by showing that the existing attacks are the result of these algebraic properties of the polynomial-based hash function. We also greatly expand the number of known weak keys and show that almost every subset of the key space is a weak-key class. The attacks presented and described in this chapter do not in any way contradict known security bounds (e.g. for GCM, as given by McGrew and Viega [196]). However, the algebraic properties (and related attacks) discussed in this chapter appear to be an inherent feature of polynomial-based authentication schemes and therefore need to be considered in a scheme's security assessment. This is demonstrated by the impact that these properties and attacks have on GCM/2<sup>+</sup> [11], a variant of GCM that we also discuss. Finally, we consider the consequences of a related property on Square Hash [105], another family of polynomial-based universal hash functions.

## 3.2 Prior work

This section gives an overview of existing proposals for polynomial-based hash functions and message authentication schemes, as well as prior security analyses for these schemes.

### 3.2.1 Polynomial-evaluation-based hash functions

The most well known and widely used universal hash function family is the polynomial evaluation hash, in which an  $\epsilon$ -almost strongly universal hash function family

## 3.2 Prior work

---

is constructed using polynomial evaluation in a finite field. This construction is described independently by Bierbrauer et al. [42], Taylor [264], and den Boer [94]; Shoup [251] identifies several variants and generalisations.

The polynomial evaluation hash uses the message  $M$  to determine a polynomial  $g_M(x) \in \mathbb{K}[x]$  with constant term equal to zero; the hash key is a random element  $H \in \mathbb{K}$  and the output of the hash function is  $\mathbf{h}_H(M) = g_M(H)$ . The canonical (and most common) method to realise this scheme is to associate a message in  $\{0, 1\}^*$  with a polynomial in  $\mathbb{K}[x]$  by  $g_M(x) = \sum_{i=1}^m M_i x^i \in \mathbb{K}[x]$ , where  $M$  is parsed as  $M_1 || \dots || M_m$  with each  $M_i \in \mathbb{K}$  and  $M_m$  possibly padded. We remark that some care may be needed to ensure that the family of hash functions chosen is indeed universal when the message space contains messages of variable length; for example, the polynomial-evaluation hash used within GCM depends on the length-encoding (discussed below) to ensure its universality.

Below, we briefly describe some authentication schemes based on polynomial evaluation hash functions that are relevant to our work. Throughout this chapter we will focus on GCM for concreteness however the majority of the comments apply equally to any other hash function based on polynomial evaluation. Most of the results in this chapter apply equally to both common constructions of MACs from universal hash functions, either  $\tau = \mathbf{E}_k(N) + \mathbf{h}_H(M)$  or  $\tau = \mathbf{E}_k(\mathbf{h}_H(M))$ , as our results are based on collisions in the hash function. Where necessary it will be made clear that a remark is dependent on one of these general constructions or the specific structure of GCM. We will also discuss Square Hash [105] in this chapter: Square Hash is another family of universal hash functions based on a different set of polynomials and we defer further details to Section 3.7.

### 3.2.1.1 Galois/Counter Mode

Galois/Counter Mode (GCM) is an AEAD scheme submitted to NIST by McGrew and Viega in 2004, with the specification slightly revised in 2005 [197] (although the revision contained ‘*no normative changes [from the 2004 specification]*’). GCM combines counter mode encryption with a polynomial-evaluation-based MAC following the Encrypt–then–MAC paradigm, although the hash key is derived from the

### 3.2 Prior work

---

block cipher key. GCM is intended to be used with 128-bit block ciphers, although an option exists to use a 64-bit block cipher. The most common instantiation is AES–GCM; we now briefly described GCM as used with a 128-bit block cipher.

Galois/Counter Mode encryption takes as input: a block cipher key  $k$ , an initialisation vector  $N$ , plaintext  $P = P_1 || \dots || P_p$  and additional data  $A = A_1 || \dots || A_a$ . The IV should preferably be 96-bits long although any length is supported (see [148]); it is critical to the security of GCM that the IV is non-repeating (i.e. is a nonce) [152]. For each  $i$ ,  $|P_i| = |A_i| = 128$ , except for perhaps partial final blocks. With this input, GCM returns a ciphertext  $C = C_1 || \dots || C_p$  (the same length as the plaintext) and an authentication tag  $\tau$  which is up to 128-bits long (specific choices of tag length may be standardised, e.g. [99, p. 9]).

The plaintext is encrypted using the block cipher in counter mode, under key  $k$  with counter value starting at  $ctr_1$ . If the IV is 96 bits long the initial counter value ( $ctr_0$ ) is  $N || 0^{31}1$ , otherwise it is a polynomial-evaluation-based hash of  $N$  after zero padding (using the hash key described below). For each  $i$ ,  $ctr_i = inc(ctr_{i-1})$ , where  $inc(\cdot)$  increments the last 32 bits of its argument (modulo  $2^{32}$ ).

The authentication tag is computed from a polynomial evaluation hash (in  $\mathbb{F}_{2^{128}}$ ). The message  $M$  is parsed as 128-bit blocks (with partial final blocks zero padded) and each block is interpreted as an element of  $\mathbb{F}_{2^{128}}$ . The first block  $M_1$  encodes the length of the (unpadded) plaintext and additional data; this block will be referred to as the ‘length field’ throughout this chapter. This is followed by blocks of ciphertext  $M_2, \dots, M_{p+1} = C_p, \dots, C_1$  and then the associated data  $M_{p+2}, \dots, M_{a+p+1} = A_a, \dots, A_1$ . Note that in this description the labelling of the blocks  $M_i$  are reversed from those given in the original GCM specification as this gives a neater description of the polynomial used in evaluating the hash function. The hash key  $H$  is derived from the block cipher key:  $H = E_k(0^{128})$ . The hash function is then computed as  $h_H(M) = \sum_{i=1}^{a+p+1} M_i H^i$  (where all operations are in  $\mathbb{F}_{2^{128}}$ ). The authentication tag is given by:

$$\tau = E_k(ctr_0) \oplus h_H(M) .$$

## 3.2 Prior work

---

### 3.2.1.2 Sophie Germain Counter Mode

In 2012, Saarinen [244] observed cycling attacks against GCM and other polynomial MACs and hashes. Following this Saarinen proposed Sophie Germain Counter Mode (SGCM) [243] as a variant of GCM. SGCM differs from GCM only by the choice of field in which the hash is computed: SGCM uses  $\mathbb{F}_q$ , where  $q = 2^{128} + 12451$ , rather than  $\mathbb{F}_{2^{128}}$ , as  $\mathbb{F}_q^*$  has significantly fewer subgroups than  $\mathbb{F}_{2^{128}}^*$ . It was claimed that SGCM offers increased resistance to cycling attacks as a result of this change.

### 3.2.1.3 GCM with short multiplications

Aoki and Yasuda [11] propose GCM/2<sup>+</sup>, a variant of GCM designed to increase the efficiency in software. GCM/2<sup>+</sup> makes several changes to the GCM specification, which we discuss in greater detail in Section 3.6.1. The most significant of these changes is to evaluate the hash function using ‘*short multiplications*’ in  $\mathbb{F}_{2^{64}}$  rather than multiplications in  $\mathbb{F}_{2^{128}}$ . However, this severely impacts the security of the scheme, as we discuss in Section 3.6.2.

### 3.2.1.4 Poly1305

Poly1305–AES was specified by Bernstein in 2005 [34], although a preliminary version appears on his website<sup>1</sup> from from 2004. The inputs to Poly1305–AES are two 128-bit keys, one for AES ( $k$ ) and one for the hash ( $r$ ) which has some specific bits set to zero; a 128-bit nonce ( $N$ ); and a message (a byte string). The output of Poly1305–AES is a 128-bit authentication tag. The hash of a message is computed by evaluating a polynomial at the secret key (in  $\mathbb{F}_{2^{130-5}}$ ) and encrypting this by adding (modulo  $2^{128}$ ) the output of  $\text{AES}_k(N)$ .

The original paper describes ‘*cipher replaceability*’ as an advantage of Poly1305 [34, p. 33]: ‘*If anything does go wrong with AES, users can switch from Poly1305–AES to Poly1305–AnotherFunction, with an identical security guarantee.*’ Indeed, a recent proposal to the IETF for a new AEAD mode aims, in part, to provide

---

<sup>1</sup><http://cr.yp.to/mac.html>

## 3.2 Prior work

---

resilience against advances in AES cryptanalysis and uses Poly1305–ChaCha20 [211]; ChaCha20 is a stream cipher also designed by Bernstein [35]. We study this proposal in greater detail in Section 6.2.

### 3.2.2 Security analyses

For a polynomial-based MAC, it is well established that the probability of creating a valid (non-truncated) tag having seen a single valid (message, tag) pair is approximately  $m/|\mathbb{K}|$  where the polynomial is evaluated in  $\mathbb{K}$  and  $m$  is the length of message that the construction operates on (see, for example, [251, 196, 111, 135]). However, it is worth emphasising that in this context  $m$  is the maximum permissible message length. This is included in the original analysis of GCM [196] but is not made explicitly clear in the later papers [111, 135]. In Section 3.3.3 we will demonstrate the importance of this distinction by describing one method for forging GCM tags using a longer message than the one that was given in the valid (message, tag) pair from the tag generation oracle.

In this section, we briefly describe the main existing results on the security of polynomial-based MACs. Because GCM is the most prominent example of a message authentication scheme based on polynomial evaluation, many of these results were originally described in terms of GCM, however they can also be applied to more general polynomial-based schemes. We will show in Section 3.4 that the attacks described in this section can be realised as special cases of the properties discussed in Section 3.3. Additionally, we discuss and extend the weak-key classes identified by Handschuh and Preneel [135] and Saarinen [244] in Section 3.5.

#### 3.2.2.1 Ferguson’s short tag attack

Ferguson’s attack against GCM when short tags are used [111] begins by observing that, for truncated tags, a full collision in the hash function is not required for a collision in the authentication tag. This is because the output of the hash function is encrypted additively and only the leading bits of the hash function affect the authentication tag.

## 3.2 Prior work

---

The second observation is that, because the polynomial hash is evaluated in a field of characteristic 2, squaring is a linear operation. So, if message blocks  $M_{2^i}$  (for some  $i$ ) are altered by an adversary, this only affects the coefficient of  $H^{2^i}$  and hence the effect on the authentication tag is a linear function of  $H$ . In particular, it is possible for the adversary to alter these message blocks in a way that guarantees particular bits of the authentication tag will not change. This means that the effective length of the authentication tag is reduced and forgeries become more likely.

Ferguson also notes that once a forgery has been observed, the adversary gains information about  $H$ . With each successive forgery, more information about  $H$  is derived and the adversary is able to use this information when manipulating the  $M_{2^i}$  and increase the number of bits of the authentication tag that are guaranteed not to change. Eventually the adversary will recover the entire hash key.

### 3.2.2.2 Joux's forbidden attack

Joux's 'forbidden attack' against GCM [152] requires two messages,  $M$  and  $M'$ , that are authenticated with the same (key, IV) pair. Reusing the (key, IV) pair in GCM has the effect of reusing  $H$ ,  $k$  and  $N$ . This allows the adversary to conclude that the xor of the authentication tags is the hash of the xor of the messages:

$$\begin{aligned}\tau_M \oplus \tau_{M'} &= (\mathbf{h}_H(M) \oplus \mathbf{E}_k(N)) \oplus (\mathbf{h}_H(M') \oplus \mathbf{E}_k(N)) \\ &= \mathbf{h}_H(M) \oplus \mathbf{h}_H(M') \\ &= \mathbf{h}_H(M \oplus M') .\end{aligned}$$

As the hash is a known polynomial evaluated at  $H$  and the adversary knows  $\tau_M$ ,  $\tau_{M'}$ , and both messages, they are able to derive a polynomial that is known to have a root at  $H$  (using the notation introduced in Section 3.3, this is  $g_{M \oplus M'} \oplus \tau_M \oplus \tau_{M'}$ ). Joux suggests that by collecting pairs of messages authenticated with the same IV, an adversary could compute the greatest common divisor (GCD) of these polynomials and eventually recover the key. This attack is prevented if we only consider nonce-respecting adversaries.

## 3.2 Prior work

---

### 3.2.2.3 Handschuh and Preneel's attacks

Handschuh and Preneel [135] describe two attacks: one allows an adversary to verify a guess for the hash key, and the second recovers the hash key.

To verify a guess for the hash key, they require an authentication tag on a message of at least two blocks, so suppose that a valid authentication tag is known for  $M_1||M_2$ . Picking any  $M'_2$  and computing  $M'_1 = M_1 + (M_2 - M'_2)H^*$ , where  $H^*$  is the guessed hash key, gives a forged message  $M'_1||M'_2$ . If the guess for the key is correct then the authentication tag for  $M_1||M_2$  is also valid for  $M'_1||M'_2$ .

Their key recovery attack extends the one described by Joux, as it does not require nonce reuse. Given a valid authentication tag on a message  $M$ , the adversary performs a verification query using the same tag but a different message  $M'$ ; this message is chosen so that the polynomial defined by  $M - M'$  has many distinct roots. A successful forgery implies that the hash key is one of the roots of this polynomial and so a binary search can be conducted on those roots in order to recover the hash key. The key recovery method was initially identified by Black and Cochran [52] but extended and generalised by Handschuh and Preneel. This attack is described as infeasible by Handschuh and Preneel in the case of GCM, due to the blocksize of 128 bits, however we will show that it is precisely as feasible as Saarinen's cycling attacks, which are described in the following section.

### 3.2.2.4 Saarinen's cycling attacks

In 2012, Saarinen [244] proposed cycling attacks against GCM and other polynomial-based MACs and hashes. The key observation is that if a hash key  $H$  lies in a subgroup of order  $t$ , then  $H^t = 1 \in \mathbb{K}$  and (for any  $i, j$ ) message blocks  $M_i$  and  $M_{i+jt}$  can be swapped without changing the value of the hash.

For example (ignoring GCM's length encoding), if  $H^4 = H$  then blocks  $M_1$  and  $M_4$

## 3.2 Prior work

---

can be swapped without changing the value of the hash:

$$\begin{aligned} \mathfrak{h}_H(M_1||M_2||M_3||M_4) &= M_1 \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \oplus M_4 \cdot H^4 \\ &= (M_1 \oplus M_4) \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \\ &= M_4 \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \oplus M_1 \cdot H^4 \\ &= \mathfrak{h}_H(M_4||M_2||M_3||M_1) . \end{aligned}$$

The attack is carried out by obtaining a valid tag for a message and performing a verification query using the same tag with the message formed by simply swapping the position of two message blocks. Saarinen observes that any  $t$  dividing  $2^{128} - 1$  can be used and that swapping  $M_i$  and  $M_{i+t}$  will give a successful forgery with probability at least  $\frac{t+1}{2^{128}}$ , as the forgery is successful if  $H$  is any one of the  $t$  elements in the chosen subgroup or if  $H$  is zero.

Saarinen [244] also describes ‘*targeted multiple bit forgeries*’ in which, instead of whole blocks, only some bits are swapped (subject to the condition that  $M_i \oplus M_{i+t}$  remains constant). This method permits the adversary more control over the forged message and the corresponding plaintext; we discuss this idea further in Section 3.3.2.

Saarinen’s main motivation for proposing SGCM [243] (described in Section 3.2.1.2) was to prevent attacks utilising this algebraic structure.

### 3.2.2.5 Weak keys

Handschuh and Preneel [135] identify a large number of weak-key classes for a variety of constructions. In particular, they observe that  $H = 0$  is a weak hash key for GCM and other similar constructions because  $\mathfrak{h}_0(M) = 0$  for every message  $M$ .

Saarinen [244] demonstrates that there are many more weak keys than are described by Handschuh and Preneel, showing that small-order subgroups of  $\mathbb{K}^*$  are weak-key classes. The forgery technique described in the previous section is successful if the hash key is an element of a small-order subgroup with order dividing the distance between the swapped message blocks. This gives a method for identifying

### 3.3 Algebraic structure and its implications

---

whether the hash key is in that class using one valid (message, tag) pair and a single verification query; these classes of weak keys meet Handschuh and Preneel's definition of weak keys (given in Section 2.3.1).

### 3.3 Algebraic structure and its implications

This section describes the main observation that allows us to give a general forgery attack against polynomial-based MAC schemes. In particular, we observe that by working with polynomials in particular ideals it is straightforward to produce forgeries for these schemes. We go on to describe a malleability property of polynomial-based MAC schemes, use this to identify a length-extension attack against GCM, and give a general method for recovering the hash key from such a scheme. Finally, we discuss possible methods to generate polynomials with the required properties.

#### 3.3.1 A generalised forgery attack

For a family of hash functions  $\mathcal{H} = \{h_H : \{0, 1\}^* \rightarrow \{0, 1\}^n \mid H \in \mathcal{K}_{\mathcal{H}}\}$  based on polynomial evaluation with  $M$  an input string, write  $h_H(M) = g_M(H)$ , where  $g_M(x) = \sum_{i=1}^m M_i x^i \in \mathbb{K}[x]$  and  $M_i, H \in \mathbb{K}$ . Now let  $q(x) = \sum_{i=1}^r q_i x^i \in \mathbb{K}[x]$  be a polynomial with constant term zero, such that  $q(H) = 0$ . Then it follows that

$$h_H(M) = g_M(H) = g_M(H) + q(H) = g_{M+Q}(H) = h_H(M + Q),$$

where  $Q = q_1 || q_2 || \dots || q_r$  and the addition  $M + Q$  is done block-wise (the shorter message is zero-padded if required). Thus, given a polynomial  $q(x)$  with these properties, it is straightforward to construct collisions for the hash function. In particular, the elements of the ideal  $\langle x^2 - Hx \rangle$  are precisely the polynomials satisfying this requirement: an element of  $\mathbb{K}[x]$  is in this ideal if and only if it has roots at both zero and  $H$ . (Recall that the ideal generated by  $f(x)$  is defined as  $\langle f(x) \rangle = \{r(x) \cdot f(x) \mid r(x) \in \mathbb{K}[x]\} \subseteq \mathbb{K}[x]$ .)

Collisions in the hash function correspond to MAC forgeries by substituting the original message for the one that yields a collision in the hash function. These forgeries arise from collisions in the hash function and hence the messages can be

### 3.3 Algebraic structure and its implications

---

substituted without any dependence on the method or key used to encrypt the output of the hash function. This method allows an adversary to create forgeries by only modifying the message, after observing a single tuple (nonce, message, tag).

The technique described above constructs forgeries via a hash collision. For this reason, the polynomial  $q(x)$  that is used to forge will always have  $x$  as a factor. This is necessary because  $g_M(x)$  is defined to have a zero constant term: if the constant term in  $g_M(x)$  was non-zero and the hash of a message was encrypted additively (i.e.  $\tau = \mathbf{E}_k(N) + \mathbf{h}_H(M)$ ), then flipping the same bits in the first message block and the authentication tag would create a valid forgery.

A related observation permits us to extend the set of polynomials that are suitable for use as a forgery polynomial in the case where the authentication tag is created by additively encrypting the output of the hash function. A similar result has been described independently and concurrently by Zhu, Tan, and Gong [274], who refer to an earlier version of the paper introducing these results [226].

If it is possible to predict an additive relation between the output of the hash function on two different messages then this difference is preserved by the additive encryption and an adversary can manipulate the value of the authentication tag accordingly. If the output of the hash function is encrypted using a block cipher then these relations are not preserved and so a full collision is required, as described above.

In this more general setting, we set  $I = \langle x - H \rangle$  and consider the canonical homomorphism into the quotient ring:

$$\begin{aligned} \phi : \mathbb{K}[x] &\rightarrow \mathbb{K}[x]/I \\ f(x) &\mapsto \bar{f}(x) = f(x) + I . \end{aligned}$$

We observe that it is possible to pick a canonical representative of each coset; by the remainder theorem  $\bar{f}(x) = f(H) + I$ . This homomorphism partitions  $\mathbb{K}[x]$  into  $|\mathbb{K}|$  equivalence classes, with  $f(x) \sim f'(x)$  precisely when  $f(H) = f'(H)$ .

Now, let  $q(x) = q_0 + q_1x + \dots + q_r x^r \in \mathbb{K}[x]$  and  $\tilde{Q} = q_1 || \dots || q_r$ , so  $\tilde{Q}$  is the concatenation of non-constant coefficients of  $q(x)$ .

### 3.3 Algebraic structure and its implications

---

Note that

$$\begin{aligned}
\phi\left(q_0 + g_{M+\tilde{Q}}(x)\right) &= \phi\left(q_0 + \sum_{i=1}^r (M_i + q_i)x^i\right) \\
&= \phi\left(\sum_{i=1}^r M_i x^i\right) + \phi\left(q_0 + \sum_{i=1}^r q_i x^i\right) \\
&= \phi(g_M(x)) + \phi(q(x)) \text{ ,}
\end{aligned}$$

and also that

$$\begin{aligned}
\phi\left(q_0 + g_{M+\tilde{Q}}(x)\right) &= \phi(q_0 + (M_1 + q_1)x + \dots + (M_r + q_r)x^r) \\
&= \phi(q_0) + \phi\left(\sum_{i=1}^r (M_i + q_i)x^i\right) \text{ .}
\end{aligned}$$

Therefore

$$\phi(g_M(x)) + \phi(q(x)) = \phi(q_0) + \phi\left(\sum_{i=1}^r (M_i + q_i)x^i\right) \text{ ,}$$

which is equivalent to the following statements:

$$\begin{aligned}
g_M(x) + q(x) + I &= q_0 + \sum_{i=1}^r (M_i + q_i)x^i + I \text{ ,} \\
\exists p(x) \in I \text{ s.t. } g_M(x) + q(x) &= q_0 + \sum_{i=1}^r (M_i + q_i)x^i + p(x) \text{ ,} \\
g_M(H) + q(H) &= q_0 + \sum_{i=1}^r (M_i + q_i)H^i + \underbrace{p(H)}_{=0} \text{ ,} \\
\mathfrak{h}_H(M) + q(H) - q_0 &= \mathfrak{h}_H(M + \tilde{Q}) \text{ .}
\end{aligned}$$

This means that if  $H$  is a root of  $q(x)$  then forging with  $\tilde{Q}$  creates a predictable difference between the hash outputs. Because  $\tau_M = \mathbf{E}_k(N) + \mathfrak{h}_H(M)$ , these predictable differences between the hash outputs will also be present as differences between the authentication tags.

The description given at the beginning of this section is a specific case of this observation in which we require that  $q(H) = q_0 = 0$ .

### 3.3 Algebraic structure and its implications

---

#### 3.3.2 Malleability

Saarinen [244] also describes ‘targeted multiple bit forgeries’ against GCM where, rather than swapping the full blocks  $M_i$  and  $M_{i+jt}$ , corresponding bits in each ciphertext block are flipped. This is a special case of the general attack, this time using a multiple of  $q(x)$ .

If  $q(H) = 0$ , then  $\alpha \cdot q(H) = 0$  for any  $\alpha \in \mathbb{K}$  and

$$\begin{aligned} \tau_M &= \mathbf{E}_k(N) + \mathbf{h}_H(M) \\ &= \mathbf{E}_k(N) + M_1 \cdot H + \cdots + M_m \cdot H^m \\ &= \mathbf{E}_k(N) + (M_1 + \alpha q_1) \cdot H + \cdots + (M_m + \alpha q_m) \cdot H^m \\ &= \tau_{M+\alpha Q} , \end{aligned}$$

where  $\tau_{M+\alpha Q}$  is the authentication tag for the message  $M_1 \oplus \alpha \cdot q_1 || \dots || M_m \oplus \alpha \cdot q_m$  (recall that  $M$  contains associated data, ciphertext, and the length of both).

If the plaintext is encrypted using a stream cipher (or a block cipher in counter mode) flipping bits in the ciphertext flips the same bits in the plaintext. This allows us to predict relations between the original plaintext and the forged plaintext (as  $C_i \oplus \alpha q_i$  decrypts to  $P_i \oplus \alpha q_i$ ). Because  $\alpha$  can be chosen so as to set  $C_i \oplus \alpha q_i$  equal to any value chosen by the adversary (for a single  $i$ ), an adversary can choose a differential between the original message and the forged message (in a single block).

If further control over the underlying plaintext is required, several forgery polynomials could be used. For example, using two forgery polynomials  $q_1$  and  $q_2$  an adversary can choose constants  $\alpha_1$  and  $\alpha_2$  and create the forgery  $M \oplus \alpha_1 q_1 \oplus \alpha_2 q_2$ . In the best case, using  $t$  polynomials permits the adversary control over  $t$  message blocks. The cost of this extra malleability is that the forgery is only successful if the hash key is a root of the greatest common divisor of the two polynomials. This can be extended to give as much control over the plaintext as required, but for every extra malleable block the success probability is reduced by at least  $\frac{1}{|\mathcal{K}_{\mathcal{H}}|}$ .

If the plaintext was encrypted using a block cipher not in counter mode then an adversary would not have this fine control over the plaintext, but would still be able to manipulate the ciphertext in this way.

### 3.3 Algebraic structure and its implications

---

This property also permits an adversary to create as many forgeries as there are non-zero elements in the field. Black and Cochran [52] and McGrew and Fluhrer [195] discuss multiple forgeries further.

#### 3.3.3 Length extension

In the GCM specification, the last block input to the hash function (corresponding to the term  $M_1 \cdot H$  in the MAC calculation) describes the length of the plaintext and additional data. The malleability property described in Section 3.3.2 allows an adversary to manipulate the length field (even though it does not explicitly appear in the sent message). If an adversary is given a valid tag for a message then the content of the length field is known, as it correctly encodes the length of the plaintext and additional data. It is therefore possible to choose a difference in the length field so that it corresponds to the length of the new message. In particular, forgeries can be created using high degree polynomial  $q(x)$  regardless of the size of the message in the initial (message,tag) pair.

This is an important remark as it removes one significant limitation on the effectiveness of cycling attacks against GCM [244], which is the length of the message necessary to launch an attack. For a cycling attack to be attempted, an adversary requires as many blocks of correctly authenticated data as there are elements in the subgroup with which he wishes to forge, in order to swap the first and last blocks. By manipulating the length field any forgery probability can be realised starting with a valid authentication tag on a single message block.

A common criticism of GCM is that the maximum message length may be restrictive in the future as data rates increase [111]. However, it follows from our work (and the original security proofs [196]) that increasing the maximum permissible length would significantly decrease the security of the scheme.

### 3.3 Algebraic structure and its implications

---

#### 3.3.4 Key recovery

Saarinen suggests that once a successful cycling attack has been carried out, the adversary would create many forgeries by further cycling attacks [244, Sect. 9]. Translating this to the more general polynomial root description: once a successful forgery occurs, the hash key is known to be one of the roots of the ‘forgery polynomial’  $q(x)$ . Therefore rather than making repeated ‘cycling forgeries’ with guaranteed success but limited control of the plaintext, the adversary can aim to recover the hash key and forge authentication tags for arbitrary messages. By attempting to forge using a subset of the roots of the forgery polynomial (and reducing the number of roots in the subset after each successful attempt), an adversary can gradually recover the hash key using a method that is independent of encryption method or key used, provided that they are willing to accept a forgery probability less than 1 at each stage. Handschuh and Preneel [135] describe choosing the subsets to realise a binary search of the key space (with a 50% forgery probability at each stage), however the adversary can choose any trade-off between the forgery probability and the speed of recovering the hash key. Additionally, as described by Joux [152], if the adversary can create many forgeries then computing the GCD of the forgery polynomials may significantly reduce the number of possible hash keys.

By testing for membership of subsets of the key space, it is plausible that an adversary could recover one bit of the hash key with each forgery attempt. If  $q(x) = \prod_{H \in \mathcal{Y}} (x - H)$ , where  $\mathcal{Y}$  is the set of hash keys for which the first bit is zero, then a successful forgery confirms that the first bit of the hash key is zero and a failure confirms that the first bit is one. Repeating this for each bit of the hash key, the whole key could be recovered using 128 verification queries.

This would require infeasibly large messages to be used in the forgery attempts if the hash keys correspond to elements of a field with  $|\mathbb{K}| \approx 2^{128}$ , but it is a strong argument against using a hash function based on polynomial evaluation in a field with  $|\mathbb{K}| \ll 2^{128}$ . This may be a direction taken by variants of GCM designed to improve performance (see [11] for one such example and Section 3.6 for an analysis of this scheme). In the case of GCM the size of the subsets that can be tested is limited to around  $2^{56}$  as the maximum message length is limited.

### 3.3 Algebraic structure and its implications

---

One advantage of being able to test for membership of arbitrary subsets is that it allows the adversary to use any partial knowledge of the hash key that they may have. Note that in the case of GCM, recovery of the hash key  $H$  does not lead to the recovery of the encryption key  $k$  as  $H = \mathbf{E}_k(0)$ .

#### 3.3.5 Choosing polynomials

For any attack utilising this algebraic structure, a critical component is the method used to select ‘good’ forgery polynomials. To maximise the probability of a successful forgery it is important that the polynomial used to attempt a forgery has many distinct roots, as a root with multiplicities increases the degree of the polynomial (and hence the length of the attempted forgery) without increasing the probability of success. The naïve way to achieve this is to compute  $q(x) = \prod_i (x - H_i)$  for as many  $H_i$  as is required to give the desired forgery probability.

Alternatively, if the polynomial defined by the hash function is evaluated in  $\mathbb{F}_{p^r}$  and the irreducible factorisation of  $x^{p^r} - x$  is computed in a subfield  $\mathbb{F}_{p^d}$ , a subset of these factors can be multiplied together (in  $\mathbb{F}_{p^d}$ ). By choosing distinct irreducible factors, the roots of the product polynomial will be distinct. Cycling attacks [244] employ a variation on this method. The factorisation

$$2^{2^n} - 1 = \prod_{i=1}^n 2^{2^{i-1}} + 1 ,$$

allows Saarinen to find factors of  $x^{2^{128}} - x$  in  $\mathbb{F}_2[x]$  which can be used in a cycling attack (although they are not necessarily irreducible):

$$\begin{aligned} x^{2^{128}} - x &= x(x-1) \frac{(x^3-1)}{x-1} \frac{(x^5-1)}{x-1} \frac{(x^{17}-1)}{x-1} \dots \\ &= x(x-1)(1+x+x^2)(1+x+\dots+x^4)(1+x+\dots+x^{16}) \dots \end{aligned}$$

To carry out the attack using a subgroup of order  $t$ , the factors  $x$ ,  $(x-1)$  and  $(1+x+\dots+x^{t-1})$  are multiplied together to obtain the polynomial  $x^{t+1} - x$ . In general there is no requirement to select  $(x-1)$  or to use only three factors, for example the polynomial  $x(1+x+x^2)(1+x+\dots+x^{16})$  could be used to give a forgery probability of  $\frac{19}{2^{128}}$ . This is not a cycling attack, as the polynomial used contains more than two terms so the forgery does not involve simply swapping two message blocks, however it does rely on the same underlying algebraic structure.

### 3.3 Algebraic structure and its implications

---

A third option is to use a randomly selected polynomial in  $\mathbb{F}_{p^r}[x]$ . One potential issue with this method is the presence of repeated factors; if a factor appears more than once in the factorisation of the forgery polynomial then the degree of the forgery polynomial (and hence the length of the forged message) is increased, without improving the success probability. Square-free factorisation has been extensively studied as it is a common first step in many polynomial factorisation algorithms (e.g. [269, Ch. 14]). It may be feasible to sample polynomials from  $\mathbb{F}_{p^r}[x]$  randomly and process this polynomial to make it more desirable by removing repeated factors. However, the presence of repeated factors is only a fairly small problem as the fraction of polynomials in  $\mathbb{F}_{p^r}$  with a repeated factor is approximately  $\frac{1}{p^r}$  [68, 216].

A larger issue is that a random polynomial in  $\mathbb{F}_{p^r}[x]$  does not necessarily split in  $\mathbb{F}_{p^r}[x]$ . Irreducible factors in  $\mathbb{F}_{p^r}[x]$  do not have roots at possible hash keys and so will never evaluate to zero and hence do not increase the success probability. It is well known that the fraction of degree  $d$  polynomials that is irreducible is approximately  $\frac{1}{d}$  [68, 185, 216], so this is not a significant problem as the forgery polynomial will probably be chosen to have a high degree in order to realise a larger success probability. However, a forgery polynomial consisting of only a few linear factors and a several high degree irreducible factors will give a low success probability and there are many polynomials in  $\mathbb{F}_{p^r}[x]$  with this form. Irreducible polynomials in  $\mathbb{F}_p$  that are known to have a root in  $\mathbb{F}_{p^r}$  would be good candidates for attempting forgeries as the normality of  $\mathbb{F}_{p^r}/\mathbb{F}_p$  guarantees that these polynomials will split into linear factors. Unfortunately this does not appear to be a well-studied area.

The main disadvantage of choosing random polynomials is that, although the roots of a polynomial in  $\mathbb{K}[x]$  can be identified efficiently (see [33] for example), it would be unlikely that a non-intersecting subset of the key space would be used for a second forgery attempt if the first was unsuccessful. This rules out utilising the key space search described in Section 3.3.4.

Bogdanov et al. [2] discuss an alternative approach for generating forgery polynomials. Their proposal resolves several limitations of the approaches discussed above: both the naïve approach and the factorisation approach require extensive pre-processing, while random polynomials are likely to have repeated roots (either within one forgery polynomial, or between several choices of forgery polynomial).

### 3.4 Algebraic structure of previous attacks

---

Using twisted polynomials in Ore rings, Bogdanov et al. are able to describe sparse forgery polynomials which partition the key space of the hash function and exist regardless of the field in which the polynomial is evaluated.

## 3.4 Algebraic structure of previous attacks

In this section, we explain the relationship between the properties described in Section 3.3 and the known results against GCM and polynomial-evaluation-hash-based MACs, which were described in Section 3.2.

### 3.4.1 Ferguson’s short tag attack

Ferguson’s attack against GCM when short tags are used [111] begins by the adversary manipulating the message in blocks  $M_{2^i}$  (for some  $i$ ). This is equivalent to attempting to forge using a linearised polynomial, that is, a polynomial  $q(x) = \sum_i q_i x^i$  for which  $q_i = 0$  unless  $i = 2^j$  for some  $j$ . Linearised polynomials have the property that their roots form a linear subspace of the splitting field of the polynomial (see [185, Ch 3.4] for an overview). Ferguson uses polynomials in  $\mathbb{F}_2[x]$  that split over  $\mathbb{F}_{2^{128}}$ , so the roots correspond to possible hash keys and this guarantees that the each root of the polynomial is distinct.

If the forgery is not successful then the adversary can choose a different (and distinct) subspace of the key space; if the forgery is successful, the adversary can choose a subspace that is contained within the original subspace. This corresponds to choosing a second forgery polynomial with either a distinct set of roots or a subset of the roots of the original forgery polynomial. Because of the structure of the roots of linearised polynomials, it is possible to describe the roots of a linearised polynomial using a matrix over  $\mathbb{F}_2$ . Multiple successful forgeries reduce the dimension of the subspace of the key space containing the hash key, which is equivalent to reducing the number of roots of the forgery polynomial; eventually an adversary will recover the key by reducing the dimension of the subspace to zero, or equivalently by reducing the degree of the forgery polynomial to one.

## 3.4 Algebraic structure of previous attacks

---

### 3.4.2 Joux's forbidden attack

Joux's 'forbidden attack' against GCM [152] is also a specific case of the properties discussed in this chapter. Joux observes that if an adversary obtains two messages that are authenticated using the same IV then they are able to derive a polynomial that is known to be satisfied by  $H$ , by finding the xor of the authentication tags and recalling the definition of the polynomial hash. The suggestion in Joux's paper is that once one polynomial has been computed, the adversary may *'hesitate between a large number of possible  $H$ '*. The proposed solution to this issue is to compute more forgery polynomials using more pairs of messages that have been authenticated with the same IVs and then to compute the GCD of all of these. We note that the techniques described in Sections 3.3.2 and 3.3.4 can also be applied once one successful forgery polynomial has been identified.

### 3.4.3 Handschuh and Preneel's attacks

Handschuh and Preneel [135] describe a method to verify a guess for a key and a key-recovery attack.

The method for verifying a key guess  $H^*$  corresponds precisely with attempting to forge using the polynomial  $x^2 - H^*x$ . A successful forgery confirms that either  $H = H^*$  or  $H = 0$ .

The key-recovery attack consists of attempting to create a forgery and then conducting a binary search through the roots of the polynomial defined by the difference between the original message and the forged message. As with Joux's forbidden attack, there is no reason why the techniques described in Sections 3.3.2 and 3.3.4 cannot be applied once one successful forgery polynomial has been identified. It is also possible to use the length-extension attack described in Section 3.3.3 to increase the forgery probability.

### 3.4 Algebraic structure of previous attacks

---

#### 3.4.4 Saarinen's cycling attacks

Saarinen observed that, if a hash key  $H$  lies in a subgroup of order  $t$ , then  $H^t = 1 \in \mathbb{K}$  and (for any  $i, j$ ) message blocks  $M_i$  and  $M_{i+jt}$  can be swapped without changing the value of the hash.

We suggest that it is more natural and general to consider the hash keys that fall in low order subgroups as roots of a low degree polynomial. Noting that  $H^{t+1} = H$  is equivalent to  $H^{t+1} - H = 0$ , we can describe cycling attacks in terms of the more general attack introduced in this chapter using the polynomial

$$q(x) = (M_i - M_{i+jt})(x^{t+1} - x) ,$$

noting that in fields of characteristic 2 subtraction is the same as  $\oplus$ .

This observation rephrases the example given in Section 3.2.2.4 as:

$$\begin{aligned} \mathbf{h}_H(M_1||M_2||M_3||M_4) &= M_1 \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \oplus M_4 \cdot H^4 \\ &= M_1 \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \oplus M_4 \cdot H^4 \\ &\quad \oplus (M_1 \oplus M_4) \cdot H \oplus (M_1 \oplus M_4) \cdot H^4 \\ &= M_4 \cdot H \oplus M_2 \cdot H^2 \oplus M_3 \cdot H^3 \oplus M_1 \cdot H^4 \\ &= \mathbf{h}_H(M_4||M_2||M_3||M_1) . \end{aligned}$$

Using the more general ‘polynomial roots’ description it is possible to forge using any subset of the key space, however if the hash keys that we wish to attempt to forge with are the elements of a low order subgroup then the polynomial that is created corresponds precisely to Saarinen’s cycling attack.

For example, the order three subgroup of  $\mathbb{F}_{2^{128}}^*$  plus the all zero key corresponds to the polynomial  $(x - H_0)(x - H_1)(x - H_2)(x - H_3) = x^4 - x$ , with the  $H_i$  as identified by Saarinen [244, Sect. 4.1]:

$$\begin{aligned} H_0 &= 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00 \\ H_1 &= 80\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00\ 00 \\ H_2 &= 10\ D0\ 4D\ 25\ F9\ 35\ 56\ E6\ 9F\ 58\ CE\ 2F\ 8D\ 03\ 5A\ 94 \\ H_3 &= 90\ D0\ 4D\ 25\ F9\ 35\ 56\ E6\ 9F\ 58\ CE\ 2F\ 8D\ 03\ 5A\ 94 . \end{aligned}$$

### 3.5 Weak keys

In this section we discuss the existing results on weak keys for polynomial-based authentication schemes and expand the known weak-key classes.

#### 3.5.1 Handschuh and Preneel’s weak key

Handschuh and Preneel [135] identify  $H = 0$  as a weak authentication key for GCM and other similar constructions because  $h_0(M) = 0$  for every message  $M$ .

Following the definition given in Section 2.3.1 and because  $|\mathcal{D}| = 1$ , an adversary is not allowed to test any key by exhaustive search, nor are they allowed any verification queries. Therefore for a single element subset of the key space  $\mathcal{D} = \{H^*\}$  to be a weak-key class, a nonce-respecting adversary must be able to identify whether or not  $H = H^*$  when they are given only (message, tag) pairs of their choosing, each created using a different IV.

We note that it is possible for a nonce-respecting adversary to detect whether  $\mathcal{D} = \{0\}$  if  $|N| \neq 96$ : in this case all values of  $N$  hash to give the same initial counter value and  $h_0(M) = 0$  for every message  $M$  so all messages have the same authentication tag (as identified in [196, Sect. 5]). However, if  $|N| = 96$  a different initial counter value is used to encrypt the output of the hash function for each tag and so, although the output of the hash function does not change, this cannot be detected given only the output of the MAC algorithm. Therefore,  $0$  is a weak key for GCM only if  $|N| \neq 96$  and is not, in general, a weak key for polynomial-evaluation-based message authentication schemes.

However, the behaviour of the zero key is highly undesirable; the value of the authentication tag depends only on the IV and not on the message. This means that, given a valid (message, tag) pair, an adversary would be able to substitute any message and still have a valid pair. This is not captured by Handschuh and Preneel’s notion of weak keys, due to fact that the adversary is not permitted any verification queries. The zero key is a ‘weaker key’ than other keys: for any key guess it is possible to construct a forgery so that it is successful if the key guess is correct (see

### 3.5 Weak keys

---

Section 3.3 and the discussion regarding cosets), but if the guess is the zero key then any forgery will be successful if the key guess is correct.

#### 3.5.2 Saarinen's weak keys

Saarinen [244] demonstrates that the situation is much worse than described by Handschuh and Preneel, describing classes of weak keys where the authentication key either falls in a low order subgroup of  $\mathbb{K}^*$  or is zero. It is then possible to create a valid forgery by swapping two message blocks of a valid (message, tag) pair without changing the authentication tag if the authentication key lies in a subgroup with order dividing the distance between the swapped message blocks.

This forgery will be successful if and only if either the key is an element of such a subgroup or is zero. Hence, this provides a simple method for identifying weak keys which requires one valid (message, tag) pair and one verification query. These classes of weak keys therefore meet Handschuh and Preneel's definition of weak keys (given in Section 2.3.1).

For example, the set of authentication keys corresponding to zero and the elements of the subgroup of order 3 in  $\mathbb{F}_{2^{128}}$  is a weak-key class. Membership of this class can be confirmed by a successful forgery if  $M_i$  and  $M_j$  are swapped and  $i \equiv j \pmod{3}$ . This is equivalent to a successful forgery using (a multiple of) the polynomial  $x^4 - x$ .

#### 3.5.3 New weak-key classes

For each subset of the key space it is possible to construct several polynomials that will evaluate to zero on any element of that set and to a non-zero field element otherwise. A successful forgery using one of these polynomials confirms that  $H$  was in the subset of the key space used to define the polynomial and a failed forgery attempt confirms that the  $H$  was not in that subset. This polynomial may not have 'nice' binary coefficients like the polynomials for Saarinen's cycling attacks but instead will, in general, be an element of  $\mathbb{F}_{2^{128}}[x]$ ; this is not problematic.

### 3.5 Weak keys

---

It follows that almost every subset of the key space for a polynomial-evaluation-based MAC is weak, regardless of the method of encryption used to form the authentication tag from the output of the hash function. Using the properties discussed in Section 3.3 with the observation above it is possible to test for membership of any subset of the key space using at most two verification queries. Membership of a subset  $\mathcal{D}$  that includes the zero key can be tested by setting  $q(x) = \prod_{H \in \mathcal{D}} (x - H)$ . This therefore requires one verification query, independent of the size of  $\mathcal{D}$ . To test for membership of a subset  $\mathcal{D}$  that does not include zero, first test whether  $H \in \mathcal{D} \cup \{0\}$  and then rule out  $H = 0$  using the method described below. This therefore requires two verification queries, but again is independent of the size of  $\mathcal{D}$ . As before, the size of the subset that can be tested is limited by the maximum message length permitted by the scheme.

The distinction between subsets including zero or not including zero is a consequence of the constant term of  $g_M(x)$  being zero to avoid predictable changes in the output of the hash from flipping low order bits. Therefore, using Handschuh and Preneel's definition, a set  $\mathcal{D}$  of hash keys for a universal-hash-based authentication scheme is a weak-key class if either:  $|\mathcal{D}| \geq 3$ , or  $|\mathcal{D}| \geq 2$  and  $0 \in \mathcal{D}$ , regardless of how  $\mathbf{h}_H(M)$  is encrypted to form  $\tau$ .

If the encryption is performed additively (as it is in GCM), then it is possible to extend this result to include every subset  $\mathcal{D}$  with  $|\mathcal{D}| \geq 2$ , using the observation in Section 3.3 regarding the quotient ring  $\mathbb{K}[x]/\langle x - H \rangle$ . This has also been noted independently by Zhu et al. [274].

Given one valid (message, tag) pair for a single block message and one verification query it is easy to determine whether or not  $H = 0$ . If the adversary attempts to forge using any other single block message and the same tag, then the forgery is successful if and only if  $H = 0$  as seen below.

### 3.6 GCM with short multiplications

---

If no length encoding is used:

$$\begin{aligned}\tau &= \mathbf{E}(ctr_0) + (M \cdot H) \\ &= \mathbf{E}(ctr_0) + (M' \cdot H) \\ &\Leftrightarrow (M - M') \cdot H = 0 \\ &\Leftrightarrow M = M' \text{ or } H = 0 .\end{aligned}$$

If a GCM style length encoding is used:

$$\begin{aligned}\tau &= \mathbf{E}(ctr_0) + (\text{length} \cdot H) + (M \cdot H^2) \\ &= \mathbf{E}(ctr_0) + (\text{length} \cdot H) + (M' \cdot H^2) \\ &\Leftrightarrow (M - M') \cdot H^2 = 0 \\ &\Leftrightarrow M = M' \text{ or } H = 0 .\end{aligned}$$

## 3.6 GCM with short multiplications

### 3.6.1 Introduction

In 2012, Yasuda and Aoki [11] proposed GCM/2<sup>+</sup>, a variant of GCM that evaluates the hash function using ‘*short multiplications*’ in  $\mathbb{F}_{2^{64}}$  rather than multiplications in  $\mathbb{F}_{2^{128}}$ . The motivation for this change is to increase the efficiency in software, where  $\mathbb{F}_{2^{128}}$  multiplications are significantly more expensive than  $\mathbb{F}_{2^{64}}$  multiplications. However this change to the specification makes the attacks in Section 3.3 much more efficient and greatly increases the number of weak keys.

The most significant difference from the GCM specification relates to the polynomial  $\bar{g}$  that is used to define the hash function  $\bar{h}$  in GCM/2<sup>+</sup>. The hash key  $H$  is split into two ‘half keys’  $H = L || R$ , where  $|L| = |R| = \frac{n}{2}$  and each message block is considered as two ‘half blocks’  $M_i = M_i^{(L)} || M_i^{(R)}$ . We will use  $M^{(L)}$  to denote  $M_1^{(L)} || \dots || M_m^{(L)}$  and similarly for  $M^{(R)}$ . The length of the additional authenticated data is encoded in  $M_1^{(L)}$  and the length of the ciphertext is encoded in  $M_1^{(R)}$ . This is identical to the GCM specification but is noteworthy because GCM/2<sup>+</sup> carries out all operations on half blocks.

### 3.6 GCM with short multiplications

---

The hash function is evaluated in two halves,  $\overline{\mathbf{h}}_H = \mathbf{h}_L || \mathbf{h}_R$ , and

$$\begin{aligned}\overline{\mathbf{h}}_H(M) &= \overline{g_M}(H) \\ &= \mathbf{h}_L(M^{(L)}) || \mathbf{h}_R(M^{(R)}) \\ &= g_{M^{(L)}}(L) || g_{M^{(R)}}(R) ,\end{aligned}$$

where  $g_{M^{(\cdot)}}(\cdot)$  is evaluated in  $\mathbb{F}_{2^{n/2}}$ . The key remark at this point is that  $\overline{\mathbf{h}}_H$  is simply the concatenation of the evaluation of two polynomials,  $g_{M^{(L)}}$  and  $g_{M^{(R)}}$ .

GCM/2<sup>+</sup> also makes the following changes to the GCM specification:

**Block size:** GCM/2<sup>+</sup> supports the use of a block cipher with any even block size (denoted by  $n$ ).

**Tag Encryption:** An extra block cipher call is added to the end of the GCM authentication tag generation algorithm. The authentication tag is computed as  $\tau = E_k(\overline{\mathbf{h}}_H(M) \oplus E_k(ctr_0))$ .

**Final Multiplication:** There is no final multiplication by the hash key in the evaluation of the hash function. The hash function polynomial is  $\overline{g_M}(H) = M_m H^{m-1} + \dots + M_2 H + M_1$ , rather than  $\overline{g_M}(H) = M_m H^m + \dots + M_2 H^2 + M_1 H$ . The requirement for the constant term of  $g_M(x)$  to be zero (in order to prevent the predictable bit flipping, as described in Section 3.3) can be relaxed due to the introduction of the tag encryption.

We will consider the half-sized multiplications and the removal of the final multiplication below. The support for other block sizes need not be considered as all of the results in this chapter are independent of the block size of the block cipher. The introduction of an extra block cipher call slightly affects our results. We also note that the specification of GCM/2<sup>+</sup> makes no recommendations regarding maximum message length.

#### 3.6.2 Attacks

As the hash function consists of the concatenation of two polynomial hash functions (each evaluated in  $\mathbb{F}_{2^{n/2}}$ ), there is no interaction between the two sides of the compu-

### 3.6 GCM with short multiplications

---

tation until the output is encrypted with the block cipher. Therefore we can consider forgery polynomials  $q^{(L)}(x)$  and  $q^{(R)}(x)$  for the two polynomial hash functions, as described in Section 3.3. Because of the tag encryption introduced in GCM/2<sup>+</sup>, we require a full collision and hence forgery polynomials with  $q_0 = 0$ ; we are unable to use the general technique described in Section 3.3 that allows us to utilise any polynomial. The forgery will be successful if  $q^{(L)}(L) = q^{(R)}(R) = 0$ . However, as there is no interaction between the two sides of the  $\overline{g_M}(H)$ , we can choose (without loss of generality)  $q^{(R)} \equiv 0$ . In this case, we have reduced finding a collision for  $\overline{h_H}$  to finding hash collisions for  $h_L$ .

If we set  $q^{(R)} \equiv 0$  then we are unable to alter the right half of any message block (in particular  $M_1^{(R)}$ ) so we cannot change the length of the ciphertext. Provided that  $q^{(L)} \not\equiv 0$ , the adversary would still have total control over the length of the additional authenticated data. Similarly, we could choose  $q^{(L)} \equiv 0$  and lose the ability to increase the length of the additional authenticated data, while retaining control of the ciphertext length. In the GCM specification the maximum length of the additional authenticated data is significantly larger than the maximum length of the ciphertext and so, if this is mimicked in the specification of GCM/2<sup>+</sup>, setting  $q^{(R)} \equiv 0$  does not significantly reduce the potential for a length extension attack.

We note that it is possible to attack both  $L$  and  $R$  simultaneously, by choosing both  $q^{(R)} \not\equiv 0$  and  $q^{(L)} \not\equiv 0$ . However, this forgery will only be successful if  $q^{(L)}(L) = q^{(R)}(R) = 0$ . As the left and right components of the hash function behave independently, the overall success probability is simply the product of the success probability for the components. Choosing  $q^{(R)} \equiv 0$  results in a success probability of 1 for the right component, which allows the adversary to identify a subset containing  $L$ , without any chance of the forgery failing due to the right half key. This leads to a faster key recovery than attacking both halves simultaneously; if an adversary can try every half key by using  $t$  queries, then they can cover the entire key space with  $2t$  queries by attacking the two halves separately whereas  $t^2$  queries are required to cover the key space if both halves are attacked together.

If the maximum length of the message is close to  $2^{64}$  blocks then we can achieve a significant success probability. Every half key is a root of  $x^{2^{64}} - x$  and therefore if messages of at least  $2^{64}$  blocks are permitted a forgery can be made with probability

### 3.6 GCM with short multiplications

---

$1 - \frac{1}{2^{64}}$  given a single valid (message, tag) pair. This probability is not quite one, as we may need to manipulate the length field (the constant term in each polynomial). It is not possible to do this using  $q(x) = x^{2^{64}} - x$ , so we will use  $x^{2^{64}-1} - 1$ . This forgery will fail only if the half key is zero. Alternatively, an adversary could recover a half key with one valid (message, tag) pair and 65 verification queries by utilising a binary search (as described in Section 3.3.4 and [135]), with an additional query at the end to decide whether or not  $H = 0$ .

Not permitting  $2^{64}$ -block messages prevents this highly efficient attack, but similar attacks are still possible. If  $m$ -block messages are permitted, it is possible to recover the key with at most  $2^{64-\log m} + \log m + 1$  verification queries. In this case,  $m$  keys can be tested with each verification query, so we will partition the key space into  $2^{64-\log m}$  subsets of size  $m$ . The attack begins by using up to  $2^{64-\log m}$  verification queries to establish which subset contains the hash key. We then conduct a binary search of the appropriate partition, requiring  $\log m$  queries and finally use one more query to establish whether or not the half key is zero. This demonstrates that the attack remains feasible if  $\log m$  is not much smaller than 64.

For example, if  $2^{56}$ -block messages are permitted (as is the case for GCM) then  $2^{56}$  keys can be tested with a single verification query. By partitioning the key space into  $2^8$  sets and using a binary search of the relevant set, at most  $2^8 + 56 + 1 \approx 315$  verification queries are required to recover a half key.

We also remark that the original GCM proposal [197] includes an appendix describing GCM with a 64-bit block cipher. In this case the polynomial evaluation is computed in  $\mathbb{F}_{2^{64}}$ . This leads to exactly the same problem as described above and the (full) hash key can be recovered using approximately 315 verification queries.

The attacks in this section highlight the relationship between the field size, maximum message length, and the forgery probability or speed of key recovery. We recommend against the use of GCM/ $2^+$  as, even if the maximum message length is a single block, it offers a worse security guarantee than GCM.

### 3.7 Square Hash

#### 3.7.1 Introduction

Square Hash was proposed by Etzel, Patel, and Ramzan in 1999 [105]. It is a universal hash function family based on MMH [132] and follows a slightly different structure than those considered so far in this chapter. We will assume that Square Hash is being used in a MAC algorithm (as described in Sections 2.2.7 and 3.2.1) and identify classes of weak keys for the hash component of the MAC algorithm.

Square Hash consists of functions  $h_k : \mathbb{Z}_p^m \rightarrow \mathbb{Z}_p$  where  $p$  is prime. For each  $k, M \in \mathbb{Z}_p^m$ ,  $h_k$  is defined by

$$h_k(M) = \sum_{i=1}^m (M_i + k_i)^2 \pmod{p} .$$

Handschuh and Preneel [135] have identified a number of weak keys for Square Hash, in particular those keys with  $k_i = k_j$  for some  $i$  and  $j$ , which can be identified by a successful forgery when  $M_i$  and  $M_j$  are swapped.

We demonstrate that every Square Hash key is an element of several weak-key classes of the form  $\mathcal{D}_{j,\mu}^{i,\lambda} = \{k \in \mathbb{Z}_p^m \mid \lambda k_i = \mu k_j\}$ , where  $\lambda, \mu \in \mathbb{Z}_p$ . We assume that an adversary can ask for a message of their choice to be authenticated and then aims to forge using a different message but the same authentication tag. All of the queries that our adversary will ask consist of just two message blocks. We will use  $M_1 || M_2$  to represent the message sent to the MAC generation oracle, and  $M'_1 || M'_2$  to represent the message sent to the verification oracle (with the authentication tag that is valid for  $M_1 || M_2$ ). The results can be trivially extended to messages consisting of several blocks provided that  $M_r = M'_r$  for all  $r \neq i, j$  and analogous methods can be applied to identify  $\mathcal{D}_{j,\mu}^{i,\lambda}$  for any  $i \neq j$  with  $i, j \leq m$ .

## 3.7 Square Hash

---

### 3.7.2 Weak-key classes

The key observation is that

$$\begin{aligned} h_k(M) &= \sum_{i=1}^m (M_i + k_i)^2 \pmod{p} \\ &= \sum_{i=1}^m M_i^2 + 2 \sum_{i=1}^m (M_i \cdot k_i) + \sum_{i=1}^m k_i^2 \pmod{p}. \end{aligned}$$

So, for a fixed key it is possible to find a hash collision (and hence a MAC forgery) if it is possible to find two messages  $M$  and  $M'$  that meet the following two conditions:

**Condition 1:**  $\sum_{i=1}^m M_i^2 = \sum_{i=1}^m M_i'^2 \pmod{p}$

**Condition 2:**  $\sum_{i=1}^m (M_i \cdot k_i) = \sum_{i=1}^m (M_i' \cdot k_i) \pmod{p}$

It is possible to identify whether or not a particular relationship holds between two key blocks (e.g.  $\lambda k_i = \mu k_j$  for some  $\lambda, \mu \in \mathbb{Z}_p$ ) using one MAC generation and one MAC verification query. As  $|\mathcal{D}_{j,\mu}^{i,\lambda}| > 1$  for every  $i, j \in \{1, \dots, m\}$ ,  $\lambda, \mu \in \mathbb{Z}_p$ , these are weak-key classes for Square Hash. We now describe two methods for identifying the two messages required to determine whether  $k \in \mathcal{D}_{2,\mu}^{1,\lambda}$  in the case where the message consists of only two blocks.

#### 3.7.2.1 Method 1

Condition 1 described above can be satisfied by choosing pairs of messages  $(M_1, M_2)$ ,  $(0, M_2')$  such that  $M_1^2 + M_2^2 = M_2'^2$ . We will test for the class of weak keys  $\mathcal{D}_{2,\mu}^{1,\lambda}$  where  $\lambda, \mu \neq 0$  using a well-known formula of Euclid.

Setting

$$\begin{aligned} M_1 &= 2\lambda\mu & M_1' &= 0 \\ M_2 &= \lambda^2 - \mu^2 & M_2' &= \lambda^2 + \mu^2, \end{aligned}$$

### 3.7 Square Hash

---

we observe that

$$\begin{aligned}
 M_1^2 + M_2^2 &= (4\lambda^2\mu^2) + (\lambda^4 - 2\lambda^2\mu^2 + \mu^4) \\
 &= \lambda^4 + 2\lambda^2\mu^2 + \mu^4 \\
 &= (\lambda^2 + \mu^2)^2 \\
 &= M_2'^2,
 \end{aligned}$$

and also that

$$\begin{aligned}
 k_1M_1 + k_2M_2 &= k_1(2\lambda\mu) + k_2(\lambda^2 - \mu^2) \\
 &= 2k_1\lambda\mu + \frac{\lambda k_1}{\mu}(\lambda^2 - \mu^2) \\
 &= k_1\left(\frac{\lambda^3}{\mu} + \lambda\mu\right) \\
 &= \frac{k_1\lambda}{\mu}(\lambda^2 + \mu^2) \\
 &= k_2M_2'.
 \end{aligned}$$

Therefore the hash of  $(M_1, M_2)$  is equal to the hash of  $(M_1', M_2')$  and the MAC forgery is successful if  $k \in \mathcal{D}_{2,\mu}^{1,\lambda}$ . It can easily be seen that a successful MAC forgery is in fact both necessary and sufficient for  $k \in \mathcal{D}_{2,\mu}^{1,\lambda}$ . Using this technique it is possible to recover the relationship between any  $k_i$  and  $k_j$  using no more than 1 valid (message, tag) pair and  $p$  verification queries.

#### 3.7.2.2 Method 2

Alternatively, we can test for membership of  $\mathcal{D}_{2,\mu}^{1,\lambda}$  (where  $\lambda \neq 0 \neq \mu$ ) following MacKay and Mahajan's preprint [190]. The two messages will be found by considering alternative factorisations of an element of  $\mathbb{Z}$  and mapping these into  $\mathbb{Z}_p$ .

As we wish to test whether  $\lambda k_1 = \mu k_2$ , take  $\omega, \tilde{\omega}$  so that  $\omega k_1 = \tilde{\omega} k_2$  and  $\omega \equiv \tilde{\omega} \pmod{2}$ . One possible method to do this is  $\omega = \lambda$  if  $\lambda \equiv \mu \pmod{2}$  and  $\omega = 2\lambda$  if  $\lambda \not\equiv \mu \pmod{2}$ , with similar definitions for  $\tilde{\omega}$ .

Now choose  $x, \tilde{x}$  so that  $x\omega = \tilde{x}\tilde{\omega}$  with  $x \equiv \tilde{x} \pmod{2}$  and  $\omega \equiv x \pmod{2}$ .

### 3.8 Discussion

---

Setting

$$\begin{aligned} M_1 &= \frac{x + \omega}{2} & M'_1 &= \frac{x - \omega}{2} \\ M_2 &= \frac{\tilde{x} - \tilde{\omega}}{2} & M'_2 &= \frac{\tilde{x} + \tilde{\omega}}{2}, \end{aligned}$$

it can be seen that Condition 1 is satisfied, as  $k_2\tilde{\omega} = k_1\omega$  and

$$\begin{aligned} M_1^2 + M_2^2 &= \left(\frac{x + \omega}{2}\right)^2 + \left(\frac{\tilde{x} - \tilde{\omega}}{2}\right)^2 \\ &= \frac{1}{4} [x^2 + 2x\omega + \omega^2 + \tilde{x}^2 - 2\tilde{x}\tilde{\omega} + \tilde{\omega}^2] \\ &= \frac{1}{4} [x^2 + \omega^2 + \tilde{x}^2 + \tilde{\omega}^2] \\ &= \frac{1}{4} [x^2 - 2x\omega + \omega^2 + \tilde{x}^2 + 2\tilde{x}\tilde{\omega} + \tilde{\omega}^2] \\ &= \left(\frac{x - \omega}{2}\right)^2 + \left(\frac{\tilde{x} + \tilde{\omega}}{2}\right)^2 \\ &= M_1'^2 + M_2'^2. \end{aligned}$$

Also, Condition 2 is satisfied:  $x\omega = \tilde{x}\tilde{\omega}$  and

$$\begin{aligned} k_1M_1 + k_2M_2 &= k_1\left(\frac{x + \omega}{2}\right) + k_2\left(\frac{\tilde{x} - \tilde{\omega}}{2}\right) \\ &= \frac{1}{2} [k_1x + k_1\omega + k_2\tilde{x} - k_2\tilde{\omega}] \\ &= \frac{1}{2} [k_1x + k_1\omega + k_2\tilde{x} - k_2\tilde{\omega} + 2(k_2\tilde{\omega} - k_1\omega)] \\ &= \frac{1}{2} [k_1x - k_1\omega + k_2\tilde{x} + k_2\tilde{\omega}] \\ &= k_1\left(\frac{x - \omega}{2}\right) + k_2\left(\frac{\tilde{x} + \tilde{\omega}}{2}\right) \\ &= k_1M_1' + k_2M_2'. \end{aligned}$$

Therefore the hash of  $(M_1, M_2)$  is equal to the hash of  $(M_1', M_2')$  and the MAC forgery is successful if  $k \in \mathcal{D}_{2,\mu}^{1,\lambda}$ .

### 3.8 Discussion

In this section, we discuss our results and some related factors that affect the security of polynomial-based MACs.

## 3.8 Discussion

---

### 3.8.1 Choice of fields

It is true that the security against cycling attacks, as presented by Saarinen, can be increased by evaluating a hash function in a field with a multiplicative group, the order of which does not have many factors. However the attack described in this chapter (of which cycling attacks are a special case) applies equally well in any finite field, so the claim that ‘*The security of polynomial-evaluation MACs against attacks of this type of attack can be determined from the factorization of the group size in a straightforward manner*’ [244, Sect. 8] is somewhat misleading.

Saarinen’s claim is valid in the sense that the factorisation of  $|\mathbb{K}| - 1$  determines the extent to which the process of computing irreducible factors will succeed; however an attack using  $\prod_{H \in \mathcal{D}} (x - H)$  will work equally well in every field. In particular, it follows from our work that the SGCM variant of GCM has the same inherent weaknesses regarding polynomial-based forgery attacks.

The size of the field has another important implication to the security of a scheme, as demonstrated in Section 3.6. It is therefore important to choose an appropriately large field in which to evaluate the polynomial, or to employ some other mechanism to ensure that multiple copies of a small field do not behave independently (as in GCM/2<sup>+</sup> [11]).

### 3.8.2 Length extension

It is unfortunate that including the length of the additional authenticated data and plaintext in the input to the hash function is not sufficient to prevent the length extension attack presented in this chapter.

In schemes that use a GCM-like length encoding, if the value of the length field were encrypted using a block cipher before being input to the hash function, it would not be possible to alter the message length as described in Section 3.3.3. However, one of the design goals of GCM was to take advantage of AES pipelining, which precludes the use of the block cipher in finalising the authentication tag.

## 3.8 Discussion

---

### 3.8.3 Malleability

Part of the reason that the algebraic structure of polynomial hashing is particularly problematic for GCM is that it gives an adversary control over the changes made to the plaintext in a forged message because addition in a field of characteristic 2 is used for both counter mode encryption and the hash function evaluation.

One way to avoid this issue is to use different operations during encryption and MAC generation. This is one possible advantage of using AES-CTR and Poly1305-AES: in this scheme the MAC is computed using addition in a prime order field while the message is encrypted using addition in a field of characteristic 2.

An alternative method to increase the difficulty for an adversary attempting to make meaningful manipulations of message is to use a mode of operation other than CTR as this may prevent the *'targeted multiple bit forgeries'* [244, Sect. 6] and the analogous forgeries in this chapter affecting the plaintext quite so directly.

GCM roughly follows the Encrypt-then-MAC paradigm, as is generally perceived to be best practice (although MAC-then-Encrypt has also been proved secure in the nonce-based AEAD setting [237]). Despite going against the perceived best practice, using a MAC-then-Encrypt approach (in addition to the changes described above) may make it harder for an adversary to create ciphertexts that correctly decrypt to a plaintext known to be related to a (plaintext,ciphertext) pair obtained from a query. We note that we have not analysed this construction and that the introduction of other weaknesses caused by making these changes has not been ruled out.

### 3.8.4 Weak keys

The weak-key classes identified in Section 3.5 cause the forgery probability to be higher than expected because an adversary can detect whether the authentication key is a member of that class and, if it is, forge with probability one.

The broader issue with polynomial-evaluation-based hash functions is that it is possible to test for membership of large subsets of the key space with only one or two

### 3.8 Discussion

---

verification queries and once an adversary has successfully confirmed membership of a subset they can either continue to forge messages or conduct a search of a much reduced key space. This is an unusual and undesirable property of a cryptosystem.

It is interesting that the two-element subsets of the key space containing zero are always weak-key classes, whereas two-element subsets not containing zero may be weak depending on how the output of the hash function is encrypted. This perhaps suggests a problem with the definition of a weak-key class.

In our opinion the observations made in this chapter are unavoidable properties of hash functions based on polynomial evaluation that result from the algebraic structure of the construction. The distinction between the two methods to encrypt the output of the hash function arises from the use of the same algebraic structure to encrypt additively and the fact that the application of a block cipher removes this structure so requires a full collision. These results are therefore not best described in terms of the number of weak keys.

The most important discussion around this issue is whether an algorithm in which almost every subset of the key space is a weak-key class is a weak algorithm or whether this is a property of the construction that, although highly undesirable, is not considered to reduce the security of the scheme to an unacceptable level. We suggest that, in the case of GCM, it is the latter; in other polynomial-based MAC schemes with different parameters it may be the former and this property must be considered carefully when designing and evaluating schemes.

Recently, Abdelraheem, Bogdanov, and Tischhauser [3] have shown that the weak-key classes identified in this chapter have severe security implications for POET [4], a submission to the CAESAR competition [79]). When POET is instantiated using the polynomial-based universal hash functions studied in this chapter, the specification requires that tests for weak-key classes are carried out (referencing Saarinen's cycling attacks). However, Abdelraheem et al. reiterate our statement that all sets of keys are weak-key classes (so testing for weak keys is meaningless) and additionally identify classes that are not covered by the proposed testing strategy, exploiting these weak keys to realise forgery attacks against POET.

# The related-key security of iterated Even–Mansour ciphers

---

## Contents

4.1	Introduction . . . . .	78
4.2	Preliminaries . . . . .	80
4.3	The RKA security of $EM^\pi[1, 1]$ . . . . .	83
4.4	The RKCPA security of $EM^\pi[1, 1, 1]$ . . . . .	99
4.5	The RKCCA security of $EM^\pi[1, 1, 1, 1]$ . . . . .	114
4.6	Discussion . . . . .	125

---

*In this chapter, we initiate the study of iterated Even–Mansour ciphers under related-key attacks (RKAs). These ciphers have been studied extensively and they are widely used due to their simplicity and security, however their RKA security has so far received little attention. We show that the simplest one-round EM cipher is strong enough to achieve non-trivial levels of RKA security even under chosen-ciphertext attacks; unfortunately this class does not include the practically relevant case of offsetting keys by constants. Two rounds reach this level under chosen-plaintext attacks and three rounds boost security to resist chosen-ciphertext attacks.*

*The work described in this chapter is joint work with Pooya Farshim and appears as [109]; it is available at [108]. The publication that this chapter is based on also includes a result on the relationship between indifferntiability and RKA security. This result is Farshim’s and so is omitted here; all other contributions are joint.*

### 4.1 Introduction

Security analyses that consider related-key attacks aim to provide guarantees about the behaviour of some cryptosystem when it is used with multiple keys that are not chosen independently of each other. As discussed in Section 2.3.2, these attacks might arise through tampering with the memory in which keys are held, or via higher-level protocol design. These attacks give the adversary much more power than the conventional single-key model; indeed, some of the earliest attacks against full AES were related-key attacks [47, 48].

Bellare and Kohno [27] initiate the theoretical study of related-key attack security. Bellare and Cash [21] describe an RKA-secure PRF and Bellare, Cash, and Miller [22] use an RKA-secure PRF to generate RKA-secure variants of several cryptographic primitives. Lucks [189] continues this approach and also discusses the idea of increasing resistance against related-key attacks by processing the key with a hash function (modelled as a random oracle).

Barbosa and Farshim [15] adopt a different approach. They study the RKA security of Feistel constructions and show that by simply reusing keys across the rounds, the 3- and 4-round Feistel constructions achieve RKA security under chosen-plaintext and chosen-ciphertext attacks respectively. Barbosa and Farshim also formalise the random-oracle transformation discussed by Lucks [189].

Since key-alternating ciphers were introduced by Daemen and Rijmen [91], they have become a popular paradigm for block cipher design. Notable examples of proposed schemes following this design include AES [90, 208], Present [59], LED [131], PRINCE [61], KLEIN [129], and Zorro [118]. Although the term ‘key-alternating cipher’ was first used with the aim of facilitating a theoretical discussion of the design of AES, the idea originates in the work of Even and Mansour [106, 107] and builds on principles dating back to Shannon [249, p. 713]. Even and Mansour sought to design the simplest block cipher possible: their proposal was a single round of the scheme shown in Figure 4.1, simply xoring a key either side of some public permutation. Rivest’s DES-X construction (proposed as a means to protect DES against brute-force attacks [161]) is also closely related to this design. In this chapter, we use the terms ‘key-alternating’ and ‘iterated Even–Mansour’ interchangeably.

## 4.1 Introduction

---

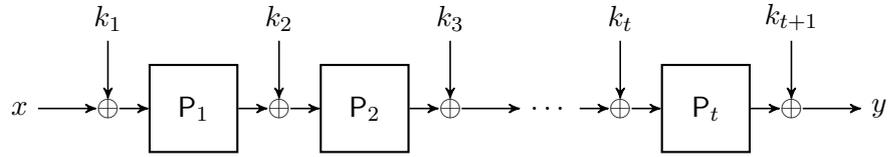


Figure 4.1: The iterated Even–Mansour scheme, where encryption is defined via  $E((k_1, \dots, k_{t+1}), x) = P_t(\dots P_2(P_1(x \oplus k_1) \oplus k_2) \dots) \oplus k_{t+1}$ .

### Contributions

Despite extensive literature on the provable security of the iterated Even–Mansour (EM) ciphers and cryptanalysis of constructions following this design strategy (including under related-key attacks), little attention has been given to the formal analysis of their related-key security. In this work we initiate the provable RKA security analysis of such ciphers. We show that the one-round EM cipher achieves a non-trivial level of RKA security under chosen-ciphertext attacks (Theorem 4.1). However, this result does not include the practically relevant case of offsetting keys by constants: we go on to show that two rounds suffice to reach this level under chosen-plaintext attacks (Theorem 4.2) and that three rounds increases security to resist chosen-ciphertext attacks (Theorem 4.6).

Our results are similar to those of Barbosa and Farshim: we show that key reuse is also a viable strategy to protect against related-key attacks in iterated Even–Mansour ciphers and use techniques similar to those used in their formalisation of the random-oracle model transform described by Lucks [189]

Starting with the simplest of the key-alternating ciphers, namely the (one-round) EM cipher, it is well known that this construction does not provide xor-RKA security (i.e. security against an adversary that can xor keys with a constant of their choosing) [61, 60, 174, 9]. We note that a similar attack prevents this construction with key-reuse from achieving xor-RKA security and describe this in greater detail in Section 4.3.1. Despite this negative result, we are able to derive conditions under which the minimal EM cipher (with key-reuse) enjoys a non-trivial level of RKA security, even in the chosen-ciphertext setting.

The search for xor-RKA security leads us to consider the two-round EM construc-

## 4.2 Preliminaries

---

tions. An offset-switching attack still applies and so again we consider key reuse. (The two permutations are still independent.) We start with chosen-plaintext attacks, formulate three new conditions (analogous to those used for the basic scheme), and prove security under them. We then show that this new set of restrictions are weak enough to follow from the standard output-unpredictability and claw-freeness properties. Since xoring with constants is output unpredictable and claw-free [27], the xor-RKA security of the single-key, two-round EM construction follows.

However, under chosen-ciphertext attacks this construction falls prey to an attack of Andreeva et al. [9] on the indistinguishability of the two-round EM cipher (adapted to the RKA setting). For CCA security, we turn to three-round constructions, where we show of the 14 possible ways to reuse keys, all but one fall prey to Andreeva et al.’s attack [9]. On the other hand, the three-round construction which uses a single key meets the desired xor-RKA security in the CCA setting.

Independently and concurrently, Cogliati and Seurin [78] also study the related-key security of iterated EM ciphers. Their Theorem 2 is very similar to our Corollary 4.7; they analyse more general key schedules and obtain tighter bounds, while our approach deals with a wider range of RKD functions.

## 4.2 Preliminaries

In this section, we briefly introduce some new notation, describe Even–Mansour ciphers, and summarise prior work analysing their security.

### 4.2.1 Notation

We briefly recall the  $\Phi$ -RKA games from Section 2.3.2. The definition of RKA security is parameterised by a set  $\Phi$ , containing functions  $\phi : \mathcal{K} \rightarrow \mathcal{K}$ . Of particular relevance to this work is the set  $\Phi^\oplus = \{k \mapsto k \oplus \Delta : \Delta \in \{0, 1\}^n\}$ . In the  $\Phi$ -RKCPA game, an adversary  $\mathcal{A}$  has access to an RK-ENC oracle. This oracle takes a pair  $(\phi, x)$ , where  $\phi \in \Phi$  and  $x$  is a point in the domain of the block cipher, returning the encryption of  $x$  under the key  $\phi(k)$ , where  $k$  is chosen uniformly at random at

## 4.2 Preliminaries

---

the beginning of the experiment and fixed throughout. The  $\Phi$ -RKCCA game adds an RK-DEC oracle which, on the same input, returns the decryption of  $x$  using the key  $\phi(k)$ . The adversary's aim is to determine whether the values that it receives from the oracles are being returned by an ideal cipher (an independent random permutation for each key) or the construction that is being reasoned about.

The adversary, construction, and RKD functions can all place queries to  $\pi$ ; we denote the number of queries made by each of these as follows: the number of distinct queries to  $\pi$  with index  $i$  made by an adversary is denoted by  $q_i$ ; the total number of queries placed to  $\pi$  by an adversary is denoted  $q_\pi = \sum_i q_i$ ; the number of distinct queries to the RK-ENC and (if present) RK-DEC oracles is denoted by  $q_{em}$ ; and the number of distinct queries to  $\pi$  with index  $i$  made by the RKD function  $\phi^\pi$  is denoted by  $q_i^\phi$ . Note that throughout this chapter,  $b$  may refer to a single bit (the challenge bit in the  $\Phi$ -RKA games from Section 2.3.2) or to a bitstring that is queried to, or returned from, an oracle; this should cause no confusion.

We call an RKA adversary repeat-free if it does not query its RK-ENC or RK-DEC oracle on a pair  $(\phi, x)$  twice. We call an RKA adversary redundancy-free if it does not query RK-ENC on  $(\phi, x)$  to get  $y$  and then RK-DEC on  $(\phi, y)$  to get  $x$ , or vice versa. Without loss of generality, all adversaries in this chapter are assumed to be both repeat-free and redundancy-free.

### 4.2.2 The Even–Mansour ciphers

The  $t$ -round Even–Mansour cipher  $\text{EM}^\pi = (\text{E}^\pi, \text{D}^\pi)$  is defined with respect to  $t$  permutations  $P_1, \dots, P_t$ . We require that each permutation has domain  $\{0, 1\}^n$ ; the resulting cipher has key space  $\mathcal{K} = \{0, 1\}^{n(t+1)}$  and domain  $\{0, 1\}^n$ , as illustrated in Figure 4.1. It is defined via

$$\begin{aligned} \text{E}^\pi((k_1, \dots, k_{t+1}), x) &= P_t(\dots P_2(P_1(x \oplus k_1) \oplus k_2) \dots) \oplus k_{t+1} , \\ \text{D}^\pi((k_1, \dots, k_{t+1}), x) &= P_1^{-1}(\dots P_{t-1}^{-1}(P_t^{-1}(x \oplus k_{t+1}) \oplus k_t) \dots) \oplus k_1 . \end{aligned}$$

In this work we are interested in EM ciphers where keys are reused in various rounds. Following notation adopted by Barbosa and Farshim [15], we denote the EM construction where key  $k_{i_j}$  is used before round  $j$  by  $\text{EM}^\pi[i_1, i_2, \dots, i_{t+1}]$ . We call such

## 4.2 Preliminaries

---

key schedules simple. Note that  $\mathcal{K} = \{0, 1\}^{n \cdot |\{i_1, i_2, \dots, i_{t+1}\}|}$  in these constructions. Of particular interest to us are  $\text{EM}^\pi[1, 1]$ ,  $\text{EM}^\pi[1, 1, 1]$ , and  $\text{EM}^\pi[1, 1, 1, 1]$  where a single key is used in all rounds. We emphasise that the round permutations in all these constructions are independently chosen, unless stated otherwise.

### 4.2.3 Security Analyses

Even and Mansour’s original analysis [106, 107] considers ‘*cracking*’ and ‘*forging*’ attacks in the random-permutation model and shows that no adversary can succeed in predicting  $x$  given  $\text{E}(k, x)$ , or in predicting  $\text{E}(k, x)$  given  $x$ , without making  $q_1$  queries to the permutation and  $q_{em}$  to the encryption/decryption oracle, where  $q_1 q_{em} \approx 2^n$ .

The indistinguishability of the Even–Mansour scheme from a random permutation is shown by Kilian and Rogaway [161, 162, Theorem 3.1 with  $\kappa = 0$ ] and Lampe, Patarin and Seurin [173, App. B of the full version]. Both works show that an adversary making  $q_{em}$  and  $q_1$  queries to the encryption/decryption oracles and the permutation oracle respectively, has a success probability of approximately  $q_1 q_{em} / 2^{n-1}$ . Gentry and Ramzan [117] show that the permutation oracle can be instantiated by a Feistel network using a random oracle without loss of security.

At Eurocrypt 2012, Dunkelman et al. [98] showed that the Even–Mansour scheme retains the same level of security using only a single key, that is  $\text{E}(k, x) = \text{P}(x \oplus k) \oplus k$ . Bogdanov et al. [60] show that the  $t$ -round Even–Mansour cipher with at least two rounds ( $t \geq 2$ ) provides security up to approximately  $2^{2n/3}$  queries and can be broken in  $t \cdot 2^{tn/(t+1)}$  queries. Following this work, several papers have moved towards proving a bound that meets this attack [258, 173], with Chen and Steinberger [73] able to prove optimal bounds using Patarin’s H-coefficient technique [217].

Chen et al. [72] consider two variants of the two-round Even–Mansour scheme: one with independent permutations and identical round keys, the other with identical permutations but a more complex key schedule. In both cases (with certain requirements on the key schedule), security is maintained up to roughly  $2^{2n/3}$  queries.

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

Additionally, Lampe and Seurin [174] show that the 12-round Even–Mansour cipher using a single key is indistinguishable from the ideal cipher. Andreeva et al. [9] show that a modification of the single-key, 5-round Even–Mansour cipher, where the key is first processed through a random oracle, is indistinguishable from the ideal cipher.

#### 4.2.4 Cryptanalysis

Daemen [88] describes a chosen-plaintext attack that recovers the key of Even–Mansour in approximately  $q_1 \approx q_{em} \approx 2^{n/2}$  queries. Biryukov and Wagner [50] are able to give a known-plaintext attack against the Even–Mansour scheme with the same complexity as Daemen’s chosen-plaintext attack. Dunkelman et al. [98] introduce the slidex attack that uses only known plaintexts and can be carried out with any number of queries provided that  $q_1 q_{em} \approx 2^n$ .

Mendel et al. [198] describe how to extend Daemen’s attack [88] to a related-key version, and are able to recover the keys when all round keys are independent. Bogdanov et al. [60] remark that related-key distinguishing attacks against the iterated Even–Mansour scheme with independent round keys ‘*exist trivially*’ and describe a key-recovery attack, requiring roughly  $2^{n/2}$  queries against the two-round Even–Mansour scheme with identical round keys, assuming that an adversary can xor constants into the round key.

Many key-alternating ciphers have been analysed in the related-key model, such as AES [47, 48], Present [215], LED [198], and Prince [150]. One of the security claims of the LED block cipher [131] is a high resistance to related-key attacks, which is justified by giving a lower bound on the number of active S-boxes.

### 4.3 The RKA security of $EM^\pi[1, 1]$

In this section we study RKD sets  $\Phi$  for which the single-key Even–Mansour construction provides  $\Phi$ -RKCCA security. Our results are similar to those of Bellare and Kohno [27], Albrecht et al. [5], and Barbosa and Farshim [15] in that we identify a set of restrictions on the RKD set  $\Phi$  that allow us to establish a security proof.

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

For the one-round Even–Mansour construction there are two simple key schedules (up to relabelling):  $EM^\pi[1, 1]$  and  $EM^\pi[1, 2]$ . Neither of these constructions can provide  $\Phi^\oplus$ -RKCPA security due to ‘offset-switching’ attacks which we describe in greater detail below. Despite this, we show that the simplest EM construction,  $EM^\pi[1, 1]$ , provides a non-trivial level of RKA security. The results of this section will also serve as a warm up to the end goal of achieving the stronger forms of RKA security discussed in later sections.

#### 4.3.1 Restricting RKD sets

We now identify and motivate conditions on a set of allowed related-key queries  $\Phi$  that allow us to argue that  $E(\phi(k), \cdot)$  and  $E(\phi'(k), \cdot)$  for  $\phi, \phi' \in \Phi$  look random and independent from an adversary’s point of view. A formal theorem statement and proof are deferred to Section 4.3.2.

As usual, our conditions impose that the RKD functions have unpredictable outputs; otherwise, RKA security is trivially unachievable as observed by Bellare and Kohno [27]. Bellare and Kohno also observe that the presence of claws in the RKD may prevent natural approaches to security proofs; our second condition excludes such RKD functions from  $\Phi$ . Our third condition is a strengthening of the claw-freeness property, motivated by preventing offset-switching attacks, which requires that it is hard to find ‘offset claws’ in  $\Phi$  for a random choice of  $k$ . (An offset claw is a pair of functions  $(\phi_1, \phi_2)$  and a value  $\Delta$  such that  $\phi_1(k) \oplus \phi_2(k) = \Delta$  with randomly chosen  $k$ .) Finally, we also consider RKD functions that depend on the underlying permutations by placing queries to them; this is particularly relevant for the Even–Mansour ciphers as they inherently operate in the random-permutation model. Our final condition places adequate restrictions on oracle queries from RKD functions to facilitate a security proof.

##### 4.3.1.1 Output unpredictability (OUP)

Bellare and Kohno [27] observe that if an adversary is able to choose  $\phi \in \Phi$  that has predictable outputs on a randomly chosen key then  $\Phi$ -RKCCA security is not

### 4.3 The RKA security of $\text{EM}^\pi[1, 1]$

---

achievable. To see this, let  $\phi$  be the constant zero (or any predictable) function. An adversary can simply test whether it is interacting with the real or the ideal cipher by enciphering  $x$  under the zero key and comparing it to the value it receives from its RK-ENC oracle on  $(\phi, x)$ . This motivates the following definition of unpredictability, adapted to the ideal-permutation model.

The advantage of an adversary  $\mathcal{A}$  against the output unpredictability (OUP) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\mathbf{Adv}_{\Phi,t}^{\text{oup}}(\mathcal{A}) = \Pr [\exists (\phi^\pi, c) \in \text{List} : \phi^\pi(k) = c : \text{List} \leftarrow_s \mathcal{A}^\pi] .$$

Here  $\text{List}$  contains pairs of the form  $(\phi^\pi, c)$  for  $\phi^\pi \in \Phi$  and  $c \in \mathcal{K}$ , and  $\pi$  is the oracle containing  $t$  ideal permutations. The probability is taken over random choices of  $k \leftarrow_s \mathcal{K}$ , the  $t$  random permutations implicit in  $\pi$ , and the coins of the adversary. Note that via a simple guessing argument, this definition can be shown to be equivalent to one where the adversary is required to output a single pair, with a loss of  $1/|\text{List}|$  in the reduction.

#### 4.3.1.2 Claw-freeness (CF)

Bellare and Kohno [27] also introduce claw-freeness (CF). Roughly speaking, a set  $\Phi$  has claws if there are two distinct  $\phi_1, \phi_2 \in \Phi$  such that  $\phi_1(k) = \phi_2(k)$ . Although this condition is not in general necessary—given an arbitrary claw there may not be an attack—the existence of claws prevents natural approaches to proofs of security. We lift claw-freeness to the ideal-permutation model.

The advantage of an adversary  $\mathcal{A}$  against the claw-freeness of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\mathbf{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{A}) = \Pr [\exists (\phi_1^\pi, \phi_2^\pi) \in \text{List} : \phi_1^\pi(k) = \phi_2^\pi(k) \wedge \phi_1^\pi \neq \phi_2^\pi : \text{List} \leftarrow_s \mathcal{A}^\pi] .$$

Here  $\text{List}$  contains pairs of RKD functions,  $\pi$  is as before, and the probability space is defined similarly to that for output unpredictability. Once again this definition is equivalent to one where  $\text{List}$  is restricted to be of size one.

### 4.3 The RKA security of $\text{EM}^\pi[1, 1]$

---

#### 4.3.1.3 Xor claw-freeness (XCF)

Claw-freeness is not a strong enough condition for the one-round EM construction to be RKA secure. Recall that for xor related-key attacks this construction does not provide RKA security due to the presence of offset-switching attacks. It is easy to check that  $\mathbb{E}((k_1, k_2), x) = \mathbb{E}((k_1 \oplus \Delta, k_2), x \oplus \Delta)$  holds with probability 1 for  $\text{EM}[1, 2]$  but only with negligible probability for the ideal cipher. One idea to thwart the above attack here would be to enforce key reuse in the construction. Although the above equality no longer holds, a close variant still applies:  $\mathbb{E}(k, x) = \mathbb{E}(k \oplus \Delta, x \oplus \Delta) \oplus \Delta$ .

This observation motivates a strengthening of the claw-freeness property requiring that it is hard to find a pair of functions  $(\phi_1, \phi_2)$  and a value  $\Delta$  such that, over a random choice of  $k$ , we have  $\phi_1(k) \oplus \phi_2(k) = \Delta$ .

The advantage of an adversary  $\mathcal{A}$  against the xor claw-freeness (XCF) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\text{Adv}_{\Phi, t}^{\text{xcf}}(\mathcal{A}) = \Pr [\exists (\phi_1^\pi, \phi_2^\pi, c) \in \text{List} : \phi_1^\pi(k) \oplus \phi_2^\pi(k) = c \wedge \phi_1^\pi \neq \phi_2^\pi : \text{List} \leftarrow_s \mathcal{A}^\pi] .$$

Here List contains tuples consisting of two RKD functions and an offset  $c \in \{0, 1\}^n$ . The probability space and  $\pi$  are defined as for claw-freeness.

Xor claw-freeness implies claw-freeness as the latter is a special case with  $c = 0$ . The fact that claw-freeness is weaker than xor claw-freeness can be seen by considering the set  $\Phi^\oplus$  corresponding to xoring with constants. This set can be easily shown to be output unpredictable and claw-free [27], but is not xor claw-free as

$$\phi_{\Delta_1}(k) \oplus \phi_{\Delta_2}(k) = \Delta_1 \oplus \Delta_2 \quad \text{where} \quad \phi_{\Delta}(k) = k \oplus \Delta .$$

We remark that the xor claw-freeness of  $\Phi$  implies at most one  $\phi \in \Phi$  is predictable: any two predictable RKD functions can be used to break xor claw-freeness.

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

#### 4.3.1.4 Xor query independence (XQI)

Let us now examine oracle access by the RKD functions. Following the attacks identified in [5, 15], we take the oracle-dependent RKD set

$$\Phi = \left\{ id : k \mapsto k, \phi^P : k \mapsto P(k) \right\} ,$$

and consider the following  $\Phi$ -RKCPA adversary against  $EM^\pi[1, 1]$ : query  $(id, 0)$  and get  $y = P(k) \oplus k$ ; query  $(\phi^P, y)$  and get  $z$ ; return  $(z = 0)$ . When interacting with  $EM^\pi[1, 1]$  we have that

$$z = E^P(P(k), P(k) \oplus k) = P(P(k) \oplus k \oplus P(k)) \oplus P(k) = P(k) \oplus P(k) = 0 .$$

On the other hand, this identity is true with probability at most  $1/(2^n - 1)$  with respect to the ideal cipher. This attack stems from the fact that when answering an RK-ENC query,  $\pi$  is evaluated at a point already queried by an RKD function; this motivates our final restriction. Informally, this condition requires that the set of values queried by RKD functions has empty intersection with the set of outputs from the RKD functions, even with offsets specified by the adversary.

The advantage of an adversary  $\mathcal{A}$  against the xor query independence (XQI) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\begin{aligned} \mathbf{Adv}_{\Phi, t}^{\text{xqi}}(\mathcal{A}) &= \Pr[\exists (i, \sigma, \phi_1^\pi, \phi_2^\pi, c) \in \text{List} : \\ &\quad (i, \phi_1^\pi(k) \oplus c, \sigma) \in \overline{\text{Qry}}[\phi_2^\pi(k)] : \text{List} \leftarrow_{\$} \mathcal{A}^\pi] , \end{aligned}$$

where

$$\begin{aligned} \text{Qry}[\phi^\pi(k)] &= \{(i, x, \sigma) : (i, x, \sigma) \text{ queried to } \pi \text{ by } \phi^\pi(k)\} , \\ \overline{\text{Qry}}[\phi^\pi(k)] &= \text{Qry}[\phi^\pi(k)] \cup \{(i, \pi(i, x, \sigma), -\sigma) : (i, x, \sigma) \in \text{Qry}[\phi^\pi(k)]\} . \end{aligned}$$

Note that for the  $EM[1, 1]$ , restricting the above definition to  $i = 1$  suffices. We also define query independence (QI) [5] as above but demand that  $c = 0^n$ .

#### 4.3.1.5 Examples

The OUP, XCF, and XQI conditions introduced above do not lead to vacuous RKD sets. As an example of an RKD set which is independent of the permutations

### 4.3 The RKA security of $\text{EM}^\pi[1, 1]$

---

consider

$$\Phi^{xu} = \{k \mapsto H(k, x) : x \in \mathcal{K}'\},$$

where  $H$  is an xor universal hash function from  $\mathcal{K}$  to  $\mathcal{K}$  with key space  $\mathcal{K}'$ . As a simple instantiation, let  $\mathcal{K}' = \{0, 1\}^k \setminus 0^k$  and for  $k \in \mathcal{K}'$  define  $H(k, x) = k \cdot x$ , where  $\{0, 1\}^k$  is interpreted as  $\text{GF}(2^k)$  with respect to a fixed irreducible polynomial and multiplication is defined over  $\text{GF}(2^k)$ .

As an example of an oracle-dependent RKD set, one can take

$$\Phi = \{k \mapsto P(k \oplus \Delta) : \Delta \in \mathcal{K}\}.$$

#### 4.3.2 Sufficiency of the conditions

We now show that if an RKD set  $\Phi$  meets the output unpredictability, xor claw-freeness and xor query independence properties defined above, then  $\text{EM}^\pi[1, 1]$  provides  $\Phi$ -RKCCA security.

**Theorem 4.1** ( $\Phi$ -RKCCA security of  $\text{EM}^\pi[1, 1]$ ). *Let  $\Phi$  be an RKD set. Then for any adversary  $\mathcal{A}$  against the  $\Phi$ -RKCCA security of  $\text{EM}^\pi[1, 1]$  with parameters as defined above, there are adversaries  $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$  and  $\mathcal{B}_4$  such that*

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1], \Phi, 1}^{\text{rkcca}}(\mathcal{A}) &\leq \text{Adv}_{\Phi, 1}^{\text{oup}}(\mathcal{B}_1) + \text{Adv}_{\Phi, 1}^{\text{xqi}}(\mathcal{B}_2) \\ &\quad + \text{Adv}_{\Phi, 1}^{\text{xcf}}(\mathcal{B}_3) + \text{Adv}_{\Phi, 1}^{\text{cf}}(\mathcal{B}_4) \\ &\quad + \frac{q_{em}(q_1 + \sum_{\phi} q_1^{\phi})}{2^n - (q_1 + \sum_{\phi} q_1^{\phi})} + \frac{2q_{em}^2}{2^n}, \end{aligned}$$

where  $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$  and  $\mathcal{B}_4$  output lists of sizes  $2q_1q_{em}, 2q_{em}^2, q_{em}^2$ , and  $q_{em}^2$  respectively and all make  $q_1$  queries to  $\pi$ .

##### 4.3.2.1 Sketch proof for Theorem 4.1

We give the intuition behind the proof here, deferring the full details to the following section. The adversary  $\mathcal{A}$  in the  $\Phi$ -RKCCA game is run with respect to the oracles

$$P(x), \quad P^{-1}(x), \quad P(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k), \quad P^{-1}(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k).$$

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

Our goal is to make a transition to an environment with the oracles

$$P(x), \quad P^{-1}(x), \quad iE(\phi^\pi(k), x), \quad iD(\phi^\pi(k), x),$$

where  $(iE, iD)$  denotes the ideal cipher. To this end, we consider two intermediate environments where the last two oracles (corresponding to RK-ENC and RK-DEC) are handled via a forgetful oracle  $\$$  that returns uniform strings on each invocation, irrespectively of its inputs. Making this change to the first environment above gives

$$P(x), \quad P^{-1}(x), \quad \$(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k), \quad \$(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k),$$

while the second gives

$$P(x), \quad P^{-1}(x), \quad \$(\phi^\pi(k), x), \quad \$(\phi^\pi(k), x),$$

both of which are identical to the environment  $(P(x), P^{-1}(x), \$(\cdot), \$(\cdot))$ . We will now argue that the above changes alter  $\mathcal{A}$ 's winning probabilities negligibly, down to the conditions on  $\Phi$  that we introduced in the previous section.

Let us first look at the change where we replace  $iE(\phi^\pi(k), x)$  and  $iD(\phi^\pi(k), x)$  with  $\$(\phi^\pi(k), x)$ . We introduce another game and replace the random keyed permutations  $iE$  and  $iD$  by random keyed functions  $iF$  and  $iC$ :

$$P(x), \quad P^{-1}(x), \quad iF(\phi^\pi(k), x), \quad iC(\phi^\pi(k), x).$$

Via (a keyed extension of) the random function/random permutation (RF/RP) switching lemma [32], the environments containing  $(iF, iC)$  and  $(iE, iD)$  can be shown to be indistinguishable up to the birthday bound  $q_{em}^2/2^n$ . The environments containing  $iF(\phi^\pi(k), x)$  and  $iC(\phi^\pi(k), x)$  and two copies of  $\$(\phi^\pi(k), x)$  and can be shown to be identical down to the CF property. Indeed, an inconsistency could arise whenever  $(\phi_1^\pi, x_1) \neq (\phi_2^\pi, x_2)$  but  $(\phi_1^\pi(k), x_1) = (\phi_2^\pi(k), x_2)$ . This means  $x_1 = x_2$  and hence we must have that  $\phi_1^\pi \neq \phi_2^\pi$ . But  $\phi_1^\pi(k) = \phi_2^\pi(k)$  and this leads to a break of the claw-freeness of  $\Phi$ .

Let us now look at the changes made when we replace  $P^\pm(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)$  with  $\$(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)$ . We need to consider the points where a forgetful simulation of  $P$  or  $P^{-1}$  via  $\$$  in the last two oracles leads to inconsistencies. Let us define the

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

following six lists:

$$\begin{aligned}
\text{List}_P^+ &= [(a, P(a)) : \mathcal{A} \text{ queries } a \text{ to } P], \\
\text{List}_P^- &= [(P^{-1}(b), b) : \mathcal{A} \text{ queries } b \text{ to } P^{-1}] , \\
\text{List}_\phi^+ &= [(a, P(a)) : \phi^\pi(k) \text{ queries } a \text{ to } P], \\
\text{List}_\phi^- &= [(P^{-1}(b), b) : \phi^\pi(k) \text{ queries } b \text{ to } P^{-1}] , \\
\text{List}_\$^+ &= [(x \oplus \phi^\pi(k), \$ (x \oplus \phi^\pi(k))) : \mathcal{A} \text{ queries } (\phi^\pi, x) \text{ to RK-ENC}] , \\
\text{List}_\$^- &= [(\$ (\phi^\pi(k) \oplus y), \phi^\pi(k) \oplus y) : \mathcal{A} \text{ queries } (\phi^\pi, y) \text{ to RK-DEC}] .
\end{aligned}$$

Let  $\text{List}_*$  be the union of the above lists over all  $\phi$  queried to RK-ENC or RK-DEC. This list encodes the trace of the attack, as in the forgetful environment no queries to  $P$  or  $P^{-1}$  are made while handling RK-ENC and RK-DEC queries.

This trace is consistent with one coming from a permutation unless  $\text{List}_*$  does not respect the permutivity properties, i.e., there are two entries  $(a, b), (a', b') \in \text{List}_*$  such that it is not the case that  $(a = a' \iff b = b')$ . Note that one of these pairs must be in  $\text{List}_\$ = \text{List}_\$^+ \cup \text{List}_\$^-$  as the other oracles are faithfully implemented. There is an inconsistency on  $\text{List}_*$  if and only if there is an inconsistency among two lists (one of which is either  $\text{List}_\$^+$  or  $\text{List}_\$^-$ ). There are 20 possibilities to consider, including the order that queries are made. We consider first query of a pair being on  $\text{List}_\$^+$ ; the other cases are dealt with symmetrically. In each case, inconsistencies can arise in two ways.

**$\text{List}_\$^+$  and  $\text{List}_P^+$**  : (1) The first component of a pair on  $\text{List}_\$^+$ —we call this a first entry on  $\text{List}_\$^+$ —matches a first entry  $a$  on  $\text{List}_P^+$ . This means that for some query  $(\phi^\pi, x)$  to RK-ENC we have that  $a = \phi^\pi(k) \oplus x$ . This leads to a break of output unpredictability. (2) The second entry on these lists match. More explicitly, we are looking at the probability that  $P(a) = R$ , for  $R$  the output of  $\$$  on a forward query. Here we can assume that  $R$  is known and this addresses the adaptivity of the choice of  $a$ . But even in this case the probability of this event is small as  $P$  is a random permutation.

**$\text{List}_\$^+$  and  $\text{List}_P^-$**  : (1) A second entry on  $\text{List}_\$^+$  matches a second entry  $b'$  on  $\text{List}_P^-$ . This means that for some query  $(\phi^\pi, x)$  to RK-ENC with output  $y$  we have that  $b' = \phi^\pi(k) \oplus y$ . This leads to a break of output unpredictability. (2) The

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

first entries match on these lists. The argument is similar to case (2) above, but now for  $P^{-1}$ .

**List $_{\S}^+$  and List $_{\phi}^+$**  : (1) A first entry on List $_{\S}^+$  matches a first entry on List $_{\phi}^+$ . This means that for some query  $(\phi_1^\pi, x)$  to RK-ENC we have that  $a = \phi_1^\pi(k) \oplus x$  for a query  $a$  of some other  $\phi_2^\pi$ . This leads to a break of xor query independence. (2) The second entries match on these lists. The argument is as in case (2) of the first pair of lists.

**List $_{\S}^+$  and List $_{\phi}^-$**  : (1) A second entry on List $_{\S}^+$  matches a second entry  $b'$  on List $_{\phi}^-$ . This means that for some query  $(\phi_1^\pi, x)$  to RK-ENC with output  $y$  we have that  $b' = \phi_1^\pi(k) \oplus y$  for a query  $b'$  of some other  $\phi_2^\pi$ . This leads to a break of xor query independence. (2) The first entries match on these lists. The argument is as in case (2) of the second pair of lists.

**List $_{\S}^+$  and List $_{\S}^+$**  : Two first entries on List $_{\S}^+$  match. This means that for two queries  $(\phi_1^\pi, x_1)$  and  $(\phi_2^\pi, x_2)$  to RK-ENC we have that  $\phi_1^\pi(k) \oplus x_1 = \phi_2^\pi(k) \oplus x_2$ . Repeat-freeness ensures that  $\phi_1 \neq \phi_2$  as otherwise  $x_1 = x_2$  as well. This leads to a break of xor claw-freeness. (2) The second entries match on these lists. Since the oracle returns independent random values, this probability can be bounded by the birthday bound.

**List $_{\S}^+$  and List $_{\S}^-$**  : A second entry on List $_{\S}^+$  matches a second entry on List $_{\S}^-$ . This means that for a queries  $(\phi_1^\pi, x_1)$  to RK-ENC with outputs  $y_1$  and  $(\phi_2^\pi, x_2)$  to RK-DEC, we have that  $\phi_1^\pi(k) \oplus y_1 = \phi_2^\pi(k) \oplus x_2$ . Redundancy-freeness ensures that  $\phi_1 \neq \phi_2$  as otherwise  $x_2$  would be an encryption of  $x_1$ . This leads to a break of xor claw-freeness. (2) The first entries match on these lists. The probability of this event can be also bounded by the birthday bound.

Hence inconsistencies among any two pairs of lists happen with small probability, and this shows that List $_{\star}$  is also inconsistent with small probability.

#### 4.3.2.2 Full details for the proof of Theorem 4.1

In this section, we give the details omitted in the proof overview of the previous section. The proof proceeds through four stages. In the first,  $\mathcal{A}$  interacts with a

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

public permutation and its inverse, plus the forward and backward directions of the Even–Mansour scheme instantiated with the same permutation:

$$P(x), \quad P^{-1}(x), \quad P(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k), \quad P^{-1}(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k) .$$

We then consider two environments in which  $P$  is replaced by  $\$$ , a forgetful random oracle, for queries made to the Even–Mansour scheme:

$$P(x), \quad P^{-1}(x), \quad \$(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k), \quad \$(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)$$

and from here we consider a keyed random function:

$$P(x), \quad P^{-1}(x), \quad iF(\phi^\pi(k), x), \quad iC(\phi^\pi(k), x) .$$

Finally, we transition to a game in which  $\$$  is replaced by an ideal cipher ( $iE, iD$ ):

$$P(x), \quad P^{-1}(x), \quad iE(\phi^\pi(k), x), \quad iD(\phi^\pi(k), x) .$$

We will now argue that the above changes alter  $\mathcal{A}$ 's winning probabilities negligibly and bound  $\mathcal{A}$ 's winning probability in terms of the conditions on  $\Phi$  introduced in Section 4.3.

The first transition is analysed via a series of games which are given in Figures 4.2, 4.3, and 4.4 and described below.

These games include two intermediate transitions: in the first,  $P$  is replaced with  $Q$  (a random permutation, chosen independently of  $P$ ) for queries arising through

<u>Game <math>i</math>:</u> $k \leftarrow_s \mathcal{K}$ $b' \leftarrow_s \mathcal{A}^{\text{RK-ENC, RK-DEC}, \pi}$ Return $b'$	<u>RK-ENC(<math>\phi^\pi, x</math>):</u> $k' \leftarrow \phi^\pi(k)$ Return $k' \oplus \text{IS}_1(k' \oplus x)$
<u><math>\pi(1, a, +)</math>:</u> Return $\text{DS}_1(a)$	<u>RK-DEC(<math>\phi^\pi, y</math>):</u> $k' \leftarrow \phi^\pi(k)$ Return $k' \oplus \text{IS}_1^{-1}(k' \oplus y)$

Figure 4.2: Procedures common to all games in the proof of Theorem 4.1. Oracles  $\pi(1, \cdot, -)$ ,  $\text{DS}_1^{-1}$ , and  $\text{IS}_1^{-1}$  are defined in a similar way to their corresponding forward oracles.

### 4.3 The RKA security of $EM^\pi[1, 1]$

<p><u>Game 1:</u></p> <p><u><math>DS_1(a)</math></u>          If <math>D_1[a] \neq \perp</math>              Return <math>D_1[a]</math>          If <math>I_1[a] \neq \perp</math>              Return <math>I_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)</math></p> <p><math>D_1[a] \leftarrow b; D_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_1) \leftarrow \text{Rng}(DS_1) \cup \{b\}</math>  <math>\text{Dom}(DS_1) \leftarrow \text{Dom}(DS_1) \cup \{a\}</math>          Return <math>D_1[a]</math></p> <p><u><math>IS_1(a)</math></u>:          If <math>I_1[a] \neq \perp</math>              Return <math>I_1[a]</math>          If <math>D_1[a] \neq \perp</math>              Return <math>D_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)</math></p> <p><math>I_1[a] \leftarrow b; I_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_1) \leftarrow \text{Rng}(IS_1) \cup \{b\}</math>  <math>\text{Dom}(IS_1) \leftarrow \text{Dom}(IS_1) \cup \{a\}</math>          Return <math>I_1[a]</math></p>	<p><u>Game 1a:</u></p> <p><u><math>DS_1(a)</math></u>:          If <math>D_1[a] \neq \perp</math>              Return <math>D_1[a]</math>          If <math>I_1[a] \neq \perp</math>              Return <math>I_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1)</math>          If <math>b \in \text{Rng}(IS_1)</math>              <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)</math>  <math>D_1[a] \leftarrow b; D_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_1) \leftarrow \text{Rng}(DS_1) \cup \{b\}</math>  <math>\text{Dom}(DS_1) \leftarrow \text{Dom}(DS_1) \cup \{a\}</math>          Return <math>D_1[a]</math></p> <p><u><math>IS_1(a)</math></u>:          If <math>I_1[a] \neq \perp</math>              Return <math>I_1[a]</math>          If <math>D_1[a] \neq \perp</math>              Return <math>D_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n</math>          If <math>b \in \text{Rng}(IS_1)</math>              <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(IS_1)</math>          If <math>b \in \text{Rng}(DS_1)</math>              <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)</math>  <math>I_1[a] \leftarrow b; I_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_1) \leftarrow \text{Rng}(IS_1) \cup \{b\}</math>  <math>\text{Dom}(IS_1) \leftarrow \text{Dom}(IS_1) \cup \{a\}</math>          Return <math>I_1[a]</math></p>
--	--

Figure 4.3: Games 1 and 1a in the proof of Theorem 4.1. Oracles  $\pi(1, \cdot, -)$ ,  $DS_1^{-1}$ , and  $IS_1^{-1}$  are defined in a similar way to their corresponding forward oracles.

### 4.3 The RKA security of $EM^\pi[1, 1]$

Game 2	Game 3:	Game 4	Game 5:
<p><u>DS<sub>1</sub>(a):</u>            If <math>D_1[a] \neq \perp</math>                Return <math>D_1[a]</math>            If <math>I_1[a] \neq \perp</math>                <math>bad_1 \leftarrow true</math>                Return <math>I_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_1)</math>            If <math>b \in Rng(IS_1)</math>                <math>bad_2 \leftarrow true</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_1, IS_1)</math>  <math>D_1[a] \leftarrow b; D_1^{-1}[b] \leftarrow a</math>  <math>Rng(DS_1) \leftarrow Rng(DS_1) \cup \{b\}</math>  <math>Dom(DS_1) \leftarrow Dom(DS_1) \cup \{a\}</math>            Return <math>D_1[a]</math></p> <p><u>IS<sub>1</sub>(a):</u>            If <math>I_1[a] \neq \perp</math>                Return <math>I_1[a]</math>            If <math>D_1[a] \neq \perp</math>                <math>bad_1 \leftarrow true</math>                Return <math>D_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n</math>            If <math>b \in Rng(IS_1)</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(IS_1)</math>            If <math>b \in Rng(DS_1)</math>                <math>bad_2 \leftarrow true</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_1, IS_1)</math>  <math>I_1[a] \leftarrow b; I_1^{-1}[b] \leftarrow a</math>  <math>Rng(IS_1) \leftarrow Rng(IS_1) \cup \{b\}</math>  <math>Dom(IS_1) \leftarrow Dom(IS_1) \cup \{a\}</math>            Return <math>I_1[a]</math></p>		<p><u>DS<sub>1</sub>(a):</u>            If <math>D_1[a] \neq \perp</math>                Return <math>D_1[a]</math>            If <math>I_1[a] \neq \perp</math>                <math>bad_1 \leftarrow true</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_1)</math>            If <math>b \in Rng(IS_1)</math>                <math>bad_2 \leftarrow true</math>  <math>D_1[a] \leftarrow b; D_1^{-1}[b] \leftarrow a</math>  <math>Rng(DS_1) \leftarrow Rng(DS_1) \cup \{b\}</math>  <math>Dom(DS_1) \leftarrow Dom(DS_1) \cup \{a\}</math>            Return <math>D_1[a]</math></p> <p><u>IS<sub>1</sub>(a):</u>            If <math>I_1[a] \neq \perp</math>                <math>bad_3 \leftarrow true</math>                Return <math>I_1[a]</math>            If <math>D_1[a] \neq \perp</math>                <math>bad_1 \leftarrow true</math>  <math>b \leftarrow_s \{0, 1\}^n</math>            If <math>b \in Rng(IS_1)</math>                <math>bad_4 \leftarrow true</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus Rng(IS_1)</math>            If <math>b \in Rng(DS_1)</math>                <math>bad_2 \leftarrow true</math>  <math>I_1[a] \leftarrow b; I_1^{-1}[b] \leftarrow a</math>  <math>Rng(IS_1) \leftarrow Rng(IS_1) \cup \{b\}</math>  <math>Dom(IS_1) \leftarrow Dom(IS_1) \cup \{a\}</math>            Return <math>I_1[a]</math></p>	

Figure 4.4: Games 2 to 5 in the proof of Theorem 4.1. Oracles  $\pi(1, \cdot, -)$ ,  $DS_1^{-1}$ , and  $IS_1^{-1}$  are defined in a similar way to their corresponding forward oracles. Boxed statements are included in Games 2 and 4, and are omitted from Games 3 and 5.

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

RK-ENC or RK-DEC; in the second,  $Q$  is replaced with  $\$$  (a forgetful random oracle). We identify the points at which these two intermediate transitions lead to inconsistencies, by setting bad flags. In contrast to how the intuition behind this proof is described in Section 4.3, we push forward the bounding of the probability bad events occurring during the first intermediate transition until after the second intermediate transition.

The specifications of  $DS_1^{-1}$ ,  $IS_1^{-1}$ , and  $\pi(\cdot, \cdot, -)$  are omitted for conciseness; they are defined analogously to their respective forward oracles. Let  $S_i$  denote the event where the adversary outputs 1 in game  $i$ .

**Game 0** is the RKA game augmented with a public permutation oracle (as described in Section 2.3.2), conditioned on  $b = 1$ . In this game, the adversary interacts with an oracle realising the public permutation  $P$  and the Even–Mansour construction instantiated with  $P$ .

**Game 1** is only syntactically different from **Game 0**. The queries to  $\pi$  are split into two groups: those made directly to  $\pi$ , either by the adversary or by an RKD function, which are answered by the sampling algorithm  $DS_1$ ; and those made indirectly, through queries made to RK-ENC, which are answered by  $IS_1$ . The oracles  $DS_1$  and  $IS_1$  maintain consistent lists  $D_1$  and  $I_1$ . The lists used by inverse oracles are identical to the lists used by the corresponding forward oracles. As this is a purely syntactic change,  $\Pr[S_0] = \Pr[S_1]$ .

**Game 1a** introduces syntactic changes to  $DS_1$  and  $IS_1$  in order for the code in the games that follow to be identical until specified bad events occur. Introducing this step allows us to remove the statement  $b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)$ ; this is necessary as we wish to completely decouple responses from  $DS_1$  and  $IS_1$ .

**Game 2** sets  $\text{bad}_1$  either if  $DS_1$  is queried on a point already defined in  $I_1$  or if  $IS_1$  is queried on a point already defined in  $D_1$  (and similarly for the inverse oracles). This occurs either because  $\mathcal{A}$  queries  $\pi$  directly at a point that is also queried to  $\pi$  through an indirect RK-ENC query, or because an RKD function queries  $\pi$  at a point that is also queried to  $\pi$  through an RK-ENC query. We will later bound the probability of this event in terms of the output unpredictability and xor query independence of  $\Phi$ . **Game 2** sets  $\text{bad}_2$  if the value chosen at random

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

for  $DS_1(a)$  is already defined in range of  $IS_1$ , or vice versa (and similarly for the inverse queries and the domain of  $IS_1$  or  $DS_1$ ). This is necessary because in Game 1, for both  $DS_1$  and  $IS_1$ ,  $b$  is sampled from  $\{0, 1\}^n \setminus \text{Rng}(DS_1, IS_1)$  whereas our objective in Game 3 is to ensure that  $DS_1$  is independent of  $IS_1$ . The outputs of  $DS_1$  and  $IS_1$  remain consistent and  $\Pr[S_1] = \Pr[S_2]$ .

**Game 3** omits the boxed statements in **Game 2** and so is identical to **Game 2** unless one of  $\text{bad}_1$  or  $\text{bad}_2$  is set. In this game, the oracles  $DS_1$  and  $IS_1$  check consistency with their own lists, but may become inconsistent with each other. It is possible for  $\text{bad}_1$  to be set in two different ways: event  $E_1$  is the event an adversary directly queries  $DS_1$  at a point coinciding with a point queried to  $IS_1$  from a query to RK-ENC (or comparable conditions resulting from queries to  $DS_1^{-1}$  or RK-DEC); event  $E_2$  is the event an RKD function queries  $DS_1$  at a point coinciding with a point queried to  $IS_1$  from a query to RK-ENC (or comparable conditions resulting from queries to  $DS_1^{-1}$  or RK-DEC). We will analyse each of the ways that  $\text{bad}_1$  can be set below. Similarly,  $\text{bad}_2$  can be set either because of a query to  $DS_1$  from  $\mathcal{A}$ , a query to  $DS_1$  from  $\phi^\pi$ , or from a query to  $IS_1$  due to a query to RK-ENC (or similarly for the corresponding inverse oracles); we consider all cases simultaneously below. In **Game 3**, the responses to RK-ENC queries are completely decoupled from the responses to  $\pi$  queries, so we can consider that RK-ENC uses  $Q$  to respond to queries and  $\pi$  uses  $P$ . We have that  $\Pr[S_2] \leq \Pr[S_3] + \Pr[E_1 \vee E_2 \vee \text{bad}_2]$ .

**Game 4** sets  $\text{bad}_3$  if  $\mathcal{A}$  queries  $IS_1$  or its inverse twice on the same point. **Game 4** chooses the response to  $IS_1$  uniformly from  $\{0, 1\}^n$  and sets  $\text{bad}_4$  if this value is already in  $\text{Rng}(IS_1)$ . The flag  $\text{bad}_4$  can be set in four ways (as a result of two queries to either of  $IS_1$  and  $IS_1^{-1}$ , plus two ‘mixed cases’ with one query to each of  $IS_1$  and  $IS_1^{-1}$ ); we consider each of these cases when we analyse the probability of setting bad events below. **Game 4** is equivalent to **Game 3** and, in particular,  $\Pr[S_3] = \Pr[S_4]$ .

**Game 5** omits the boxed statements from **Game 4** and so is identical to **Game 4** unless  $\text{bad}_3$  or  $\text{bad}_4$  is set. Let  $E'_1, E'_2, \text{bad}'_2$  represent events in **Game 5** corresponding to events  $E_1, E_2, \text{bad}_2$  in **Game 4**, then  $\Pr[E_1 \vee E_2 \vee \text{bad}_2] \leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2\Pr[\text{bad}_3 \vee \text{bad}_4]$ . In this game, calls to  $\pi(1, \cdot, +)$  through RK-ENC (which are answered by  $IS_1$ ) are answered by a forgetful random oracle and so the ciphertexts are uniform and independent of the key and the plaintext; the

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

same is true for calls to inverse oracles. Redundancy freeness guarantees that no inconsistencies arise from decrypting the result of an encryption query, or vice versa.

In Game 5, the adversary interacts with

$$P(x), \quad P^{-1}(x), \quad \$(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k), \quad \$(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k) .$$

During the next transition to

$$P(x), \quad P^{-1}(x), \quad iF(\phi^\pi(k), x), \quad iC(\phi^\pi(k), x) ,$$

inconsistencies only arise if the adversary makes queries  $(\phi_1^\pi, x_1) \neq (\phi_2^\pi, x_2)$ , but where  $(\phi_1^\pi(k), x_1) = (\phi_2^\pi(k), x_2)$ . If an adversary  $\mathcal{A}$  makes such a query, we can construct an adversary  $\mathcal{B}_4$  which wins the CF game with a list of length at most  $\frac{q_{em}^2}{2}$  as follows:  $\mathcal{B}_4$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(\phi_i^\pi, \phi_j^\pi) : 1 \leq i < j \leq q_{em}\}$ .

Considering the final transition, we switch from a random function to a random permutation (for each  $\phi^\pi$ ); the probability of an inconsistency arising in this step is bounded by  $\frac{q_{em}^2}{2^n}$  [32].

Therefore we have that

$$\mathbf{Adv}_{EM^\pi[1,1],\Phi,t}^{\text{rkcca}}(\mathcal{A}) \leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2 \Pr[\text{bad}_3 \vee \text{bad}_4] + \mathbf{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n} ,$$

and it remains to bound the probability that **bad** events occur in Game 5.

**Event  $E'_1$**  occurs when the adversary directly queries  $\pi$  at a point that is also queried as a result of a query to RK-ENC. This situation is described in Section 4.3 as an inconsistency between  $\text{List}_P$  and  $\text{List}_\Phi$ . We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_1$  against the OUP game with a list of length  $2q_1q_{em}$ . The adversary  $\mathcal{B}_1$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(\phi_i^\pi, x_i \oplus a_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_1\} \cup \{(\phi_i^\pi, y_i \oplus b_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_1\}$ , where  $x_i$  is the input to RK-ENC resulting in output  $y_i$  on the  $i^{\text{th}}$  query (reversed for a query to RK-DEC) and  $a_j$  is the input to  $\pi(1, \cdot, +)$  resulting in output  $b_j$  on the  $j^{\text{th}}$  query (similarly reversed for a query to  $\pi(1, \cdot, -)$ ). If  $\mathcal{A}$  sets  $\text{bad}_1$  with an RK-ENC or  $\text{DS}_1$  query, then  $\mathcal{B}_1$  wins the OUP game with a tuple of the form

### 4.3 The RKA security of $EM^\pi[1, 1]$

---

$(\phi_i^\pi, x_i \oplus a_j)$  and if  $\mathcal{A}$  sets  $\text{bad}_1$  with a query to  $DS_1^{-1}$  or  $IS_1^{-1}$  then  $\mathcal{B}$  wins the OUP game with a tuple of the form  $(\phi_i^\pi, y_i \oplus b_j)$ . We therefore conclude that  $\Pr[E'_1] \leq \text{Adv}_{\Phi, t}^{\text{oup}}(\mathcal{B}_1)$ , where  $\mathcal{B}_1$  outputs a list of length  $2q_1q_{em}$ .

**Event  $E'_2$**  occurs when an RKD function queries  $\pi$  at a point that is also queried as a result of a query to RK-ENC. This situation is described in Section 4.3 as an inconsistency between  $\text{List}_\phi$  and  $\text{List}_s$ . We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_2$  against the XQI game with a list of length  $2q_{em}^2$ . The adversary  $\mathcal{B}_2$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(1, +, \phi_i^\pi, \phi_j^\pi, x_i) : 1 \leq i, j \leq q_{em}\} \cup \{(1, -, \phi_i^\pi, \phi_j^\pi, y_i) : 1 \leq i, j \leq q_{em}\}$ . If  $\mathcal{A}$  sets  $\text{bad}_1$  with a query to  $IS_1$  or an RKD function that queries  $\pi(1, \cdot, +)$ , then  $\mathcal{B}_1$  wins the XQI game with a tuple of the form  $(1, +, \phi_i^\pi, \phi_j^\pi, x_i)$  and if  $\mathcal{A}$  sets  $\text{bad}_1$  with a query to  $IS_1^{-1}$  or  $\pi(1, \cdot, -)$  then  $\mathcal{B}$  wins the XQI game with a tuple of the form  $(1, -, \phi_i^\pi, \phi_j^\pi, y_i)$ . Therefore,  $\Pr[E'_2] \leq \text{Adv}_{\Phi, t}^{\text{xqi}}(\mathcal{B}_2)$ , where  $\mathcal{B}_2$  outputs a list of length  $2q_{em}^2$ .

**Flag  $\text{bad}'_2$**  is set with probability at most  $\frac{(q_1 + q_1^\phi)q_{em}}{2^n - (q_1 + \sum_\phi q_1^\phi)}$  (when it is set via a call to  $DS_1$  or its inverse); it can be set via a call to  $IS_1$  or its inverse with probability  $\frac{(q_1 + q_1^\phi)q_{em}}{2^n}$ . This situation is described in Section 4.3 as an inconsistency between  $\text{List}_p$  and  $\text{List}_s$  or  $\text{List}_\phi$  and  $\text{List}_s$ . It can be set in one of 16 different ways. Collisions between  $DS_1$  and  $IS_1$ ,  $DS_1^{-1}$  and  $IS_1^{-1}$ ,  $DS_1$  and  $IS_1^{-1}$ , or  $DS_1^{-1}$  and  $IS_1$  can all set  $\text{bad}_2$  and each is counted twice, depending on the order of the queries. This gives 8 ways to set  $\text{bad}_2$ , however the query to  $DS_1$  can arise through a query by  $\mathcal{A}$  or through  $\phi^\pi$ , which gives 16 ways. In each case, we use a birthday-bound style argument, noting that each pair  $(x, a)$  has at most a  $\frac{1}{2^n - (q_1 + \sum_\phi q_1^\phi)}$  chance of setting  $\text{bad}'_2$ ; applying the union bound and recalling that  $q_{em}$  is the total number of queries made to RK-ENC and RK-DEC (and thus to  $IS$  and  $IS_1^{-1}$ ) by  $\mathcal{A}$  (and similarly for  $q_1$  and  $q_1^\phi$ ) gives the claimed probability.

**Flag  $\text{bad}_3$**  is set if two queries to either  $IS_1$  or its inverse result in the same value being either input to or output from  $IS_1$  or its inverse. This situation is described in Section 4.3 as an inconsistency between  $\text{List}_s$  and  $\text{List}_s$ . We will use  $\mathcal{A}$  that sets  $\text{bad}_3$  to create an adversary  $\mathcal{B}_3$  against the XCF game. The adversary  $\mathcal{B}_3$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(\phi_i^\pi, \phi_j^\pi, x_i \oplus x_j) : 1 \leq i < j \leq q_{em}\} \cup \{(\phi_i^\pi, \phi_j^\pi, y_i \oplus y_j) : 1 \leq i < j \leq q_{em}\}$ . If  $\mathcal{A}$  sets  $\text{bad}_3$  as a result of a query to  $IS_1$ , then  $\mathcal{B}_3$  wins the XCF game with a tuple of the form  $(\phi_i^\pi, \phi_j^\pi, x_i \oplus x_j)$  and if  $\mathcal{A}$  sets  $\text{bad}_3$  as a result a query to  $IS_1^{-1}$ , then  $\mathcal{B}_3$  wins the XCF game

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

with a tuple of the form  $(\phi_i^\pi, \phi_j^\pi, y_i \oplus y_j)$ . Thus  $\Pr[\text{bad}_3] \leq \mathbf{Adv}_{\Phi, t}^{\text{xcf}}(\mathcal{B}_3)$ , where  $\mathcal{B}_3$  outputs a list of length at most  $q_{em}^2$ .

**Flag**  $\text{bad}_4$  is set with probability at most  $\frac{q_{em}^2}{2} \frac{1}{2^n}$  using similar reasoning as in the setting of  $\text{bad}_2$ . This situation is described in Section 4.3 as an inconsistency between  $\text{List}_\S$  and  $\text{List}_\S$ .

As we have that

$$\begin{aligned} \mathbf{Adv}_{\text{EM}^\pi[1,1], \Phi, t}^{\text{rkcca}}(\mathcal{A}) &\leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2 \Pr[\text{bad}_3 \vee \text{bad}_4] \\ &\quad + \mathbf{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, \end{aligned}$$

we may conclude that

$$\begin{aligned} \mathbf{Adv}_{\text{EM}^\pi[1,1], \Phi, t}^{\text{rkcca}}(\mathcal{A}) &\leq \mathbf{Adv}_{\Phi, t}^{\text{oup}}(\mathcal{B}_1) + \mathbf{Adv}_{\Phi, t}^{\text{xqi}}(\mathcal{B}_2) + \frac{q_{em}(q_1 + \sum_\phi q_1^\phi)}{2^n - (q_1 + \sum_\phi q_1^\phi)} \\ &\quad + 2 \left( \mathbf{Adv}_{\Phi, t}^{\text{xcf}}(\mathcal{B}_3) + \frac{q_{em}^2}{2^{n+1}} \right) + \mathbf{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, \end{aligned}$$

where  $\mathcal{B}_1$  outputs a list of length  $2q_1q_{em}$ ,  $\mathcal{B}_2$  a list of length  $2q_{em}^2$ ,  $\mathcal{B}_3$  a list of length  $q_{em}^2$ , and  $\mathcal{B}_4$  a list of length at most  $q_{em}^2$ .  $\blacksquare$

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

The theorem established in the previous section does not encompass the  $\Phi^\oplus$  set as this set is not xor claw-free. In this section, we investigate whether an extra round can boost RKA security to the  $\Phi^\oplus$  set.

For two-round EM constructions, up to relabelling there are 5 simple key schedules:  $[1, 1, 1]$ ,  $[1, 1, 2]$ ,  $[1, 2, 1]$ ,  $[1, 2, 2]$ , and  $[1, 2, 3]$ . It is easy to see that offset-switching attacks can be used to attack the  $\Phi^\oplus$ -RKCPA security of all but the first of these. In the following sections we study the  $\Phi^\oplus$ -RKA security of the only remaining construction,  $\text{EM}^\pi[1, 1, 1]$ .

## 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

### 4.4.1 Weakening the conditions

We start by following a similar proof strategy to that given for  $\text{EM}^\pi[1, 1]$  and identify a set of restrictions which are strong enough to enable a security proof, yet weak enough to encompass the  $\Phi^\oplus$  set. These conditions decouple the queries made to the permutation oracle and allow us to simulate the  $\text{P}_2$  oracle forgetfully in a reduction.

Starting from the environment

$$\pi(i, x, \sigma), \quad \text{P}_2(\text{P}_1(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) ,$$

we simulate the  $\text{P}_2$  oracle forgetfully and move to a setting with oracles

$$\pi(i, x, \sigma), \quad \$(\text{P}_1(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) \quad \equiv \quad \pi(i, x, \sigma), \quad \$() .$$

From here it is straightforward to reach the ideal game  $\pi(i, x, \sigma), \text{iE}(\phi^\pi(k), x)$  via an application of the RF/RP switching lemma [32] and the claw-freeness property as in the analysis of  $\text{EM}^\pi[1, 1]$ .

We now analyse the probability that the second environment simulates the first one in an inconsistent way. We look at inconsistencies which arise due to oracles being queried on the same inputs and derive conditions that preclude these from occurring.

#### 4.4.1.1 First-order output unpredictability (OUP1)

The first place such an inconsistency might arise is when  $\mathcal{A}$  makes an explicit  $\pi$  query  $(2, a, +)$  that matches a query made to  $\$,$  i.e.  $\text{P}_1(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k) = a$  for some  $(\phi^\pi, x)$  queried to  $\$.$  We now address this event, giving a slight strengthening of the condition as we will be using it later.

Let  $t \geq 1.$  The advantage of an adversary  $\mathcal{A}$  against the first-order output unpredictability (OUP1) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\begin{aligned} \text{Adv}_{\Phi, t}^{\text{oup1}}(\mathcal{A}) &= \Pr[\exists (i, \sigma, \phi^\pi, x, c) \in \text{List} : \\ &\quad \text{P}_i^\sigma(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k) = c : \text{List} \leftarrow_{\$} \mathcal{A}^\pi] . \end{aligned}$$

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

Oracle  $\pi$ , the probability space, and  $\text{List}$  are defined analogously to the previous definitions. Note that in the RKCPA setting we do not need to consider inconsistencies resulting from inputs to  $\text{P}_1^{-1}$  through RK-ENC queries, and thus only need to consider  $(i, \sigma) = (1, +)$  above.

##### 4.4.1.2 First-order claw-freeness (CF1)

Inconsistencies may also arise as a result of two RK-ENC queries. As  $\$$  is forgetful, a consistent simulation requires that no two RK-ENC queries query  $\$$  at the same point. This leads to the following modification of claw-freeness.

Let  $t \geq 1$ . The advantage of an adversary  $\mathcal{A}$  against the first-order claw-freeness (CF1) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\begin{aligned} \text{Adv}_{\Phi, t}^{\text{cf1}}(\mathcal{A}) &= \Pr[\exists (i, \sigma, \phi_1^\pi, x_1, \phi_2^\pi, x_2) \in \text{List} : \\ &\quad \text{P}_i^\sigma(\phi_1^\pi(k) \oplus x_1) \oplus \phi_1^\pi(k) = \text{P}_i^\sigma(\phi_2^\pi(k) \oplus x_2) \oplus \phi_2^\pi(k) \wedge \phi_1^\pi \neq \phi_2^\pi : \\ &\quad \text{List} \leftarrow_{\$} \mathcal{A}^\pi] . \end{aligned}$$

##### 4.4.1.3 First-order query independence (QI1)

We now look at inconsistencies in the simulation due to a mismatch in an RKD query to  $\pi$  and a query to  $\$$  made via the RK-ENC oracle. Since only the second function is forgetfully simulated, we require independence of queries for  $\text{P}_2$  only. Once again, in the RKCPA setting, restricting the definition to  $(i, \sigma) = (1, +)$  suffices.

Let  $t \geq 2$ . The advantage of an adversary  $\mathcal{A}$  against the first-order query independence (QI1) of an RKD set  $\Phi$  with access to  $t$  ideal permutations is defined via

$$\begin{aligned} \text{Adv}_{\Phi, t}^{\text{qi1}}(\mathcal{A}) &= \Pr[\exists (i, \sigma, \phi_1^\pi, x_1, \phi_2^\pi) \in \text{List} : \\ &\quad (2, \text{P}_i^\sigma(\phi_1^\pi(k) \oplus x_1) \oplus \phi_1^\pi(k), \pm) \in \overline{\text{Qry}}[\phi_2^\pi(k)] : \text{List} \leftarrow_{\$} \mathcal{A}^\pi] , \end{aligned}$$

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

where, as before,

$$\begin{aligned} \text{Qry}[\phi^\pi(k)] &= \{(i, x, \sigma) : (i, x, \sigma) \text{ queried to } \pi \text{ by } \phi^\pi(k)\} , \\ \overline{\text{Qry}}[\phi^\pi(k)] &= \text{Qry}[\phi^\pi(k)] \cup \{(i, \pi(i, x, \sigma), -\sigma) : (i, x, \sigma) \in \text{Qry}[\phi^\pi(k)]\} . \end{aligned}$$

#### 4.4.2 Sufficiency of the conditions

The new set of conditions identified above allow us to use a similar proof strategy to that of Theorem 4.1 and establish the following result.

**Theorem 4.2** ( $\Phi$ -RKCPA security of  $\text{EM}^\pi[1, 1, 1]$ ). *Let  $\Phi$  be an RKD set. Then for any adversary  $\mathcal{A}$  against the  $\Phi$ -RKCPA security of  $\text{EM}^\pi[1, 1, 1]$  with parameters as defined before there are  $\mathcal{B}_{1a}$ ,  $\mathcal{B}_{1b}$ ,  $\mathcal{B}_{2a}$ ,  $\mathcal{B}_{2b}$ ,  $\mathcal{B}_3$ , and  $\mathcal{B}_4$  such that*

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1],\Phi,2}^{\text{rkcpa}}(\mathcal{A}) &\leq \text{Adv}_{\Phi,2}^{\text{oup1}}(\mathcal{B}_{1a}) + \text{Adv}_{\Phi,2}^{\text{oup}}(\mathcal{B}_{1b}) \\ &\quad + \text{Adv}_{\Phi,2}^{\text{qi1}}(\mathcal{B}_{2a}) + \text{Adv}_{\Phi,2}^{\text{xqi}}(\mathcal{B}_{2b}) \\ &\quad + 2\text{Adv}_{\Phi,2}^{\text{cf1}}(\mathcal{B}_3) + \text{Adv}_{\Phi,2}^{\text{cf}}(\mathcal{B}_4) \\ &\quad + \frac{q_{em}(q_2 + \sum_\phi q_2^\phi)}{2^n - (q_2 + \sum_\phi q_2^\phi)} + \frac{2q_{em}^2}{2^n} , \end{aligned}$$

where  $\mathcal{B}_{1a}$  and  $\mathcal{B}_{1b}$  output lists of length  $q_2q_{em}$ ,  $\mathcal{B}_{2a}$  and  $\mathcal{B}_{2b}$  lists of length  $q_{em}^2$ ,  $\mathcal{B}_3$  a list of length  $q_{em}^2$ , and  $\mathcal{B}_4$  a list of length at most  $q_{em}^2$ .

The proof follows a similar pattern to the proof of Theorem 4.1 and again proceeds through four stages. In the first,  $\mathcal{A}$  interacts with the public permutations and their inverses, plus the forward direction of the 2-round Even–Mansour scheme instantiated with the same permutations:

$$\pi(i, x, \sigma), \quad \text{P}_2(\text{P}_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) .$$

We then consider an environment in which  $\text{P}_2$  is replaced by  $\mathcal{S}$ , a forgetful random oracle, for queries made to the Even–Mansour scheme:

$$\pi(i, x, \sigma), \quad \mathcal{S}(\text{P}_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k),$$

and from here we consider a keyed random function:

$$\pi(i, x, \sigma), \quad \text{iF}(\phi^\pi(k), x) .$$

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

---

Finally, we transition to a game in which  $iF$  is replaced by an ideal cipher  $iE$ :

$$\pi(i, x, \sigma), \quad iE(\phi^\pi(k), x) .$$

We will now argue that the above changes alter  $\mathcal{A}$ 's winning probabilities negligibly and bound  $\mathcal{A}$ 's winning probability in terms of the conditions on  $\Phi$  described in Sections 4.3.1 and 4.4.1.

The first transition is analysed via a series of games, given in Figures 4.5, 4.6, and 4.7. These games include two intermediate transitions: in the first,  $P_2$  is replaced with  $Q$  (a random permutation, chosen independently of  $\pi$ ) for queries arising through RK-ENC or RK-DEC; in the second,  $Q$  is replaced with  $\$$  (a forgetful random oracle). We identify the points at which these two intermediate transitions lead to inconsistencies, by setting **bad** flags. As before, let  $S_i$  denote the event where the adversary outputs 1 in game  $i$ .

<p><u>Game <math>i</math>:</u>  <math>k \leftarrow_s \mathcal{K}</math>  <math>b' \leftarrow_s \mathcal{A}^{\text{RK-ENC}, \pi}</math>            Return <math>b'</math></p> <p><u>RK-ENC(<math>\phi^\pi, x</math>):</u>  <math>k' \leftarrow \phi^\pi(k)</math>  <math>z_1 \leftarrow S_1(k' \oplus x)</math>            Return <math>k' \oplus IS_2(k' \oplus z_1)</math></p> <p><u><math>\pi(2, a, +)</math>:</u>            Return <math>DS_2(a)</math></p>	<p><u><math>\pi(1, a, +)</math>:</u>            Return <math>S_1(a)</math></p> <p><u><math>S_1(a)</math>:</u>            If <math>S_1[a] \neq \perp</math>                Return <math>S_1[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(S_1)</math>  <math>S_1[a] \leftarrow b; S_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(S_1) \leftarrow \text{Rng}(S_1) \cup \{b\}</math>  <math>\text{Dom}(S_1) \leftarrow \text{Dom}(S_1) \cup \{a\}</math>            Return <math>S_1[a]</math></p>
---	--

Figure 4.5: Procedures common to all games in the proof of Theorem 4.2. Oracles  $\pi(i, \cdot, -)$ ,  $S_i^{-1}$ , and  $DS_2^{-1}$  are defined in a similar way to their corresponding forward oracles.

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

<u>Game 1:</u>	<u>Game 2</u> <u>Game 3:</u>
<p><u>DS<sub>2</sub>(a):</u>            If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math>            If <math>I_2[a] \neq \perp</math>                Return <math>I_2[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2)</math>            If <math>b \in \text{Rng}(IS_2)</math>                  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math>  <math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_2) \leftarrow \text{Rng}(DS_2) \cup \{b\}</math>  <math>\text{Dom}(DS_2) \leftarrow \text{Dom}(DS_2) \cup \{a\}</math>            Return <math>D_2[a]</math></p> <p><u>IS<sub>2</sub>(a):</u>            If <math>I_2[a] \neq \perp</math>                Return <math>I_2[a]</math>            If <math>D_2[a] \neq \perp</math>                Return <math>D_2[a]</math>  <math>b \leftarrow_s \{0, 1\}^n</math>            If <math>b \in \text{Rng}(IS_2)</math>                  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(IS_2)</math>            If <math>b \in \text{Rng}(DS_2)</math>                  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math>  <math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_2) \leftarrow \text{Rng}(IS_2) \cup \{b\}</math>  <math>\text{Dom}(IS_2) \leftarrow \text{Dom}(IS_2) \cup \{a\}</math>            Return <math>I_2[a]</math></p>	<p><u>DS<sub>2</sub>(a):</u>            If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math>            If <math>I_2[a] \neq \perp</math>                <math>\text{bad}_1 \leftarrow \text{true};</math> <span style="border: 1px solid black; padding: 2px;">Return <math>I_2[a]</math></span>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2)</math>            If <math>b \in \text{Rng}(IS_2)</math>                <math>\text{bad}_2 \leftarrow \text{true}</math>                <span style="border: 1px solid black; padding: 2px;"><math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math></span>  <math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_2) \leftarrow \text{Rng}(DS_2) \cup \{b\}</math>  <math>\text{Dom}(DS_2) \leftarrow \text{Dom}(DS_2) \cup \{a\}</math>            Return <math>D_2[a]</math></p> <p><u>IS<sub>2</sub>(a):</u>            If <math>I_2[a] \neq \perp</math>                Return <math>I_2[a]</math>            If <math>D_2[a] \neq \perp</math>                <math>\text{bad}_1 \leftarrow \text{true};</math> <span style="border: 1px solid black; padding: 2px;">Return <math>D_2[a]</math></span>  <math>b \leftarrow_s \{0, 1\}^n</math>            If <math>b \in \text{Rng}(IS_2)</math>                  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(IS_2)</math>            If <math>b \in \text{Rng}(DS_2)</math>                <math>\text{bad}_2 \leftarrow \text{true}</math>                <span style="border: 1px solid black; padding: 2px;"><math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math></span>  <math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_2) \leftarrow \text{Rng}(IS_2) \cup \{b\}</math>  <math>\text{Dom}(IS_2) \leftarrow \text{Dom}(IS_2) \cup \{a\}</math>            Return <math>I_2[a]</math></p>

Figure 4.6: Games 1 to 3 in the proof of Theorem 4.2. Oracles  $\pi(i, \cdot, -)$ ,  $S_i^{-1}$ , and  $DS_2^{-1}$  are defined in a similar way to their corresponding forward oracles. Boxed statements are included in Game 2 and are omitted from Game 3.

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

---

Game 4	Game 5:
<p><math>DS_2(a)</math>:</p> <p>If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math></p> <p>If <math>I_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_1 \leftarrow true</math></p> <p><math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_2)</math></p> <p>If <math>b \in Rng(IS_2)</math></p> <p style="padding-left: 20px;"><math>bad_2 \leftarrow true</math></p> <p><math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math></p> <p><math>Rng(DS_2) \leftarrow Rng(DS_2) \cup \{b\}</math></p> <p><math>Dom(DS_2) \leftarrow Dom(DS_2) \cup \{a\}</math></p> <p>Return <math>D_2[a]</math></p> <p><math>IS_2(a)</math>:</p> <p>If <math>I_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_3 \leftarrow true; \boxed{\text{Return } I_2[a]}</math></p> <p>If <math>D_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_1 \leftarrow true</math></p> <p><math>b \leftarrow_s \{0, 1\}^n</math></p> <p>If <math>b \in Rng(IS_2)</math></p> <p style="padding-left: 20px;"><math>bad_4 \leftarrow true</math></p> <p style="padding-left: 40px;"><math>\boxed{b \leftarrow_s \{0, 1\}^n \setminus Rng(IS_2)}</math></p> <p>If <math>b \in Rng(DS_2)</math></p> <p style="padding-left: 20px;"><math>bad_2 \leftarrow true</math></p> <p><math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math></p> <p><math>Rng(IS_2) \leftarrow Rng(IS_2) \cup \{b\}</math></p> <p><math>Dom(IS_2) \leftarrow Dom(IS_2) \cup \{a\}</math></p> <p>Return <math>I_2[a]</math></p>	

Figure 4.7: Games 4 and 5 in the proof of Theorem 4.2. Oracles  $\pi(i, \cdot, -)$ ,  $S_i^{-1}$ , and  $DS_2^{-1}$  are defined in a similar way to their corresponding forward oracles. Boxed statements are included in Game 4 and are omitted from Game 5.

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

---

**Game 0** is the RKA game augmented with a public permutation oracle (as described in Section 2.3.2), conditioned on  $b = 1$ . In this game, the adversary interacts with an oracle realising the public permutations  $\pi$  and the forward direction of the Even–Mansour construction instantiated with  $\pi$ .

**Game 1** is only syntactically different from **Game 0**. The queries to  $\pi$  are split into three groups. The first group is those made to  $\pi(1, \cdot, \cdot)$ , either by the adversary, by an RKD function, or as the result of an RK-ENC query; these are answered by the sampling algorithms  $S_1$  and  $S_1^{-1}$ . The second group of queries consists of those made directly to  $\pi(2, \cdot, \cdot)$ , either by the adversary or by an RKD function, which are answered by the sampling algorithms  $DS_2$  and  $DS_2^{-1}$ . The third group of queries are those queries to  $\pi(2, \cdot, +)$  which are made indirectly, through queries made to RK-ENC; these queries are answered by  $IS_2$ . The oracles  $DS_2$  and  $IS_2$  maintain consistent lists  $D_2$  and  $I_2$ . As this is a purely syntactic change,  $\Pr[S_0] = \Pr[S_1]$ .

**Game 2** sets  $\text{bad}_1$  either if  $DS_2$  is queried on a point already defined in  $I_2$  or if  $IS_2$  is queried on a point already defined in  $D_2$  (and similarly for  $DS_2^{-1}$ ). This occurs either because  $\mathcal{A}$  queries  $\pi(2, \cdot, \cdot)$  directly at a point that is also queried to  $\pi(2, \cdot, \cdot)$  through an indirect RK-ENC query, or because an RKD function queries  $\pi(2, \cdot, \cdot)$  at a point that is also queried to  $\pi(2, \cdot, \cdot)$  through an RK-ENC query (and similarly for  $DS_2^{-1}$ ). We will later bound the probability of this event in terms of the output unpredictability, first-order output unpredictability, xor query independence and first-order query independence of  $\Phi$ . **Game 2** sets  $\text{bad}_2$  if the value chosen at random for  $DS_2(a)$  is already defined in range of  $IS_2$ , or vice versa (and similarly for the inverse queries and the domain of  $IS_1$ ). This is necessary because in **Game 1**, for both  $DS_2$  and  $IS_2$ ,  $b$  may be sampled from  $\{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)$  whereas our objective in **Game 3** is to ensure that responses from  $DS_2$  are independent of responses from  $IS_2$ . The code of  $S_1$  and is unchanged and will remain so throughout this proof. The outputs of  $DS_2$  and  $IS_2$  remain consistent and  $\Pr[S_1] = \Pr[S_2]$ .

**Game 3** omits the boxed statements in **Game 2** and so is identical to **Game 2** unless one of  $\text{bad}_1$  or  $\text{bad}_2$  is set. In this game, the oracles  $DS_2$  and  $IS_2$  check consistency with their own lists, but may become inconsistent with each other. It is possible for  $\text{bad}_1$  to be set in two possible ways: event  $E_1$  is the event an adversary directly queries  $DS_2$  (or  $DS_2^{-1}$ ) at a point coinciding with a point

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

---

queried to (or output from)  $IS_2$  via a query to RK-ENC; event  $E_2$  is the event an RKD function queries  $DS_2$  (or  $DS_2^{-1}$ ) at a point coinciding with a point queried to (or output from)  $IS_2$  from a query to RK-ENC. We will analyse each of the ways that  $bad_1$  can be set below. Similarly,  $bad_2$  can be set either because of a query to  $DS_2$  from  $\mathcal{A}$ , a query to  $DS_2$  from  $\phi^\pi$ , or from a query to  $IS_2$  due to a query to RK-ENC (or similarly for the corresponding inverse oracles); we consider all cases simultaneously below. In **Game 3**, the responses to RK-ENC queries are completely decoupled from the responses to  $\pi$  queries, so we can consider that RK-ENC uses  $Q$  to respond to queries and  $\pi$  uses  $P$ . We have that  $\Pr[S_2] \leq \Pr[S_3] + \Pr[E_1 \vee E_2 \vee bad_2]$ .

**Game 4** sets  $bad_3$  if two distinct queries to RK-ENC result in the same value being queried to  $IS_2$ . As  $\mathcal{A}$  makes no queries to RK-DEC, we only need to consider the possibility that  $bad_3$  is set as a result of a query to RK-ENC. **Game 4** chooses the response to  $IS_2$  uniformly from  $\{0, 1\}^n$  and sets  $bad_4$  if this value is already in  $\text{Rng}(IS_2)$ . **Game 4** is equivalent to **Game 3** and, in particular,  $\Pr[S_3] = \Pr[S_4]$ .

**Game 5** omits the boxed statements from **Game 4** and so is identical to **Game 4** unless  $bad_3$  or  $bad_4$  is set. Let  $E'_1, E'_2, bad'_2$  represent events in **Game 5** corresponding to events  $E_1, E_2, bad_2$  in **Game 4**, then  $\Pr[E_1 \vee E_2 \vee bad_2] \leq \Pr[E'_1 \vee E'_2 \vee bad'_2] + 2\Pr[bad_3 \vee bad_4]$ . In this game, calls to  $\pi(2, \cdot, +)$  through RK-ENC (which are answered by  $IS_1$ ) are answered by a forgetful random oracle and so the ciphertexts are uniform and independent of the key and the plaintexts.

In **Game 5**, the adversary interacts with

$$\pi(i, x, \sigma), \quad \$(P_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) .$$

During the next transition to

$$\pi(i, x, \sigma), \quad iF(\phi^{P, P^{-1}}(k), x) ,$$

inconsistencies only arise if the adversary makes queries  $(\phi_1^\pi, x_1) \neq (\phi_2^\pi, x_2)$ , but where  $(\phi_1^\pi(k), x_1) = (\phi_2^\pi(k), x_2)$ . If an adversary  $\mathcal{A}$  makes such a query, we can construct an adversary  $\mathcal{B}_4$  which wins the CF game with a list of length at most  $\frac{q_{em}^2}{2}$  as follows:  $\mathcal{B}_4$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(\phi_i^\pi, \phi_j^\pi) : 1 \leq i < j \leq q_{em}\}$ .

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

Considering the final transition, we switch from a random function to a random permutation (for each  $\phi^\pi$ ); the probability of an inconsistency arising in this step is bounded by  $\frac{q_{em}^2}{2^n}$  [32].

Therefore we have that

$$\begin{aligned} \mathbf{Adv}_{\text{EM}^\pi[1,1,1],\Phi,t}^{\text{rkcpa}}(\mathcal{A}) &\leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2 \Pr[\text{bad}_3 \vee \text{bad}_4] \\ &\quad + \mathbf{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}. \end{aligned}$$

It remains to bound the probability that bad events occur in Game 5.

**Event  $E'_1$**  occurs when the adversary directly queries  $\pi(2, \cdot, \cdot)$  at a point that is also queried as a result of a query to RK-ENC. Although  $\mathcal{A}$  makes no  $\text{IS}_2^{-1}$  queries, it is possible to trigger  $E'_1$  with a query to  $\text{DS}_2^{-1}$ . We will use  $\mathcal{A}$  to create adversaries  $\mathcal{B}_{1a}$  and  $\mathcal{B}_{1b}$  against the OUP1 and OUP games respectively, both with lists of length  $q_2 q_{em}$ . The adversary  $\mathcal{B}_{1a}$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, a_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_2\}$ . The adversary  $\mathcal{B}_{1b}$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(\phi_i^\pi, y_i \oplus b_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_2\}$ . If  $\mathcal{A}$  can set bad by querying the permutation at a point that is also queried as a result of a query to RK-ENC, then either  $\mathcal{B}_{1a}$  wins the OUP1 game or  $\mathcal{B}_{1b}$  wins the OUP game. We therefore conclude that  $\Pr[E'_1] \leq \mathbf{Adv}_{\Phi,t}^{\text{oup1}}(\mathcal{B}_{1a}) + \mathbf{Adv}_{\Phi,t}^{\text{oup}}(\mathcal{B}_{1b})$ , where  $\mathcal{B}_{1a}$  and  $\mathcal{B}_{1b}$  both output a list of length  $q_2 q_{em}$ .

**Event  $E'_2$**  occurs when an RKD function queries the  $\text{P}_2$  at a point that is also queried as a result of a query to RK-ENC. We will use  $\mathcal{A}$  to create adversaries  $\mathcal{B}_{2a}$  and  $\mathcal{B}_{2b}$  against the QI1 and XQI games respectively, both with lists of length  $q_{em}^2$ . The adversary  $\mathcal{B}_{2a}$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, \phi_j^\pi) : 1 \leq i, j \leq q_{em}\}$ ; the adversary  $\mathcal{B}_{2b}$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(2, -, \phi_i^\pi, \phi_j^\pi, y_i) : 1 \leq i, j \leq q_{em}\}$ . If  $\mathcal{A}$  can set bad by causing an RKD function to query the permutation at a point that is also queried as a result of a query to RK-ENC, then either  $\mathcal{B}_{2a}$  wins the QI1 game or  $\mathcal{B}_{2b}$  wins the XQI game. Although  $\mathcal{A}$  makes no  $\text{IS}_2^{-1}$  queries, it is possible to trigger  $E'_2$  with a query to  $\text{DS}_2^{-1}$  and so we must include tuples of the form  $(2, -, \phi_i^\pi, \phi_j^\pi, y_i)$ . Therefore we can conclude that  $\Pr[E'_2] \leq \mathbf{Adv}_{\Phi,t}^{\text{qi1}}(\mathcal{B}_{2a}) + \mathbf{Adv}_{\Phi,t}^{\text{xqi}}(\mathcal{B}_{2b})$ , where  $\mathcal{B}_{2a}$  and  $\mathcal{B}_{2b}$  both output lists of length  $q_{em}^2$ .

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

**Flag**  $\text{bad}'_2$  is set with probability at most  $\frac{q_{em}(q_2 + \sum_\phi q_2^\phi)}{2^n - (q_2 + \sum_\phi q_2^\phi)}$ . It can be set in one of 8 different ways (this differs from the proof of Theorem 4.1, as this theorem only considers CPA adversaries, so  $\mathcal{A}$  cannot use RK-DEC to set  $\text{bad}'_2$ ). Collisions between  $\text{DS}_2$  and  $\text{IS}_2$  or  $\text{DS}_1^{-1}$  and  $\text{IS}_2$  can set  $\text{bad}_2$ ; both of these cases are counted four times, depending on the order of the queries and whether the  $\text{DS}_2$  query was triggered by  $\mathcal{A}$  or by  $\phi^\pi$ . In each case, we use a birthday-bound style argument, noting that a pair  $(x, a)$  has at most a  $\frac{1}{2^n - (q_1 + \sum_\phi q_1^\phi)}$  chance of setting  $\text{bad}'_2$ . Applying the union bound and recalling that  $q_2$  is the total number of queries made to  $\pi(2, \cdot, \cdot)$  by  $\mathcal{A}$  gives the claimed probability.

**Flag**  $\text{bad}_3$  is set when two queries to RK-ENC result in the  $\text{P}_2$  being queried at the same point. We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_3$  against the CF1 property of  $\Phi$ . The adversary  $\mathcal{B}_3$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, \phi_j^\pi, x_j) : 1 \leq i < j \leq q_{em}\}$ . Note that, as  $\mathcal{A}$  makes no RK-DEC queries, they are unable to set  $\text{bad}_3$  with an RK-DEC query. If this were not the case, we would require that  $\Phi$  is XCF (the XCF adversary would use tuples of the form  $(\phi^\pi, y \oplus b)$ ). Thus  $\Pr[\text{bad}_3] \leq \text{Adv}_{\Phi, t}^{\text{cf1}}(\mathcal{B}_3)$ , where  $\mathcal{B}_3$  outputs a list of length  $\frac{q_{em}(q_{em}-1)}{2}$ .

**Flag**  $\text{bad}_4$  is set with probability at most  $\frac{q_{em}^2}{2} \frac{1}{2^n}$  using similar reasoning as in the setting of  $\text{bad}_2$ .

As we have that

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1], \Phi, t}^{\text{rkcpa}}(\mathcal{A}) &\leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2 \Pr[\text{bad}_3 \vee \text{bad}_4] \\ &\quad + \text{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, + \frac{q_{em}^2}{2^n}, \end{aligned}$$

we may conclude that

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1], \Phi, t}^{\text{rkcpa}} &\leq \text{Adv}_{\Phi, t}^{\text{oup1}}(\mathcal{B}_{1a}) + \text{Adv}_{\Phi, t}^{\text{oup}}(\mathcal{B}_{1b}) \\ &\quad + \text{Adv}_{\Phi, t}^{\text{qi1}}(\mathcal{B}_{2a}) + \text{Adv}_{\Phi, t}^{\text{xqi}}(\mathcal{B}_{2b}) \\ &\quad + \frac{q_{em}(q_2 + \sum_\phi q_2^\phi)}{2^n - (q_2 + \sum_\phi q_2^\phi)} + 2 \left( \text{Adv}_{\Phi, t}^{\text{cf1}}(\mathcal{B}_3) + \frac{q_{em}^2}{2^{n+1}} \right) \\ &\quad + \text{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, \end{aligned}$$

where  $\mathcal{B}_{1a}$  and  $\mathcal{B}_{1b}$  output lists of length  $q_2 q_{em}$ ,  $\mathcal{B}_{2a}$  and  $\mathcal{B}_{2b}$  lists of length  $q_{em}^2$ ,  $\mathcal{B}_3$  a list of length  $q_{em}^2$ , and  $\mathcal{B}_4$  a list of length at most  $q_{em}^2$ .  $\blacksquare$

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

##### 4.4.3 $\Phi^\oplus$ -RKA Security

We now show that the restrictions identified above are weak enough so that the offset RKD set  $\Phi^\oplus$  can be shown to satisfy them. We start by showing that for oracle-independent sets,  $\Phi$  is output unpredictable and claw-free if and only if it is first-order output unpredictable and first-order claw-free.

**Proposition 4.3** ( $\text{OUP} \wedge \text{CF} \iff \text{OUP1} \wedge \text{CF1}$ ). *Let  $\Phi$  be an oracle-independent RKD set and let  $t \geq 1$ . Then for any adversary  $\mathcal{A}$  against the OUP game outputting a list of size  $\ell$  and placing  $q_i$  permutation queries with index  $i$ , there is an adversary  $\mathcal{B}_1$  outputting a list of size  $\ell$  and placing  $q_i + \delta_{1i}\ell$  permutation queries with index  $i$  such that*

$$\text{Adv}_{\Phi,t}^{\text{oup}}(\mathcal{A}) \leq \text{Adv}_{\Phi,t}^{\text{oup1}}(\mathcal{B}_1) .$$

*Additionally, for an adversary  $\mathcal{A}$  against the CF game with the same parameters as the previous adversary, there is an adversary  $\mathcal{B}_2$  outputting a list of size  $\ell$  and placing  $q_i$  permutation queries with index  $i$ , such that*

$$\text{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{A}) \leq \text{Adv}_{\Phi,t}^{\text{cf1}}(\mathcal{B}_2) .$$

*Moreover, for any adversary  $\mathcal{A}$  against OUP1 with parameters as before, there is an adversary  $\mathcal{B}_1$  against OUP outputting a list of size  $\ell \cdot q_\pi = \ell \sum_i q_i$ , where it places  $q_i$  permutation queries with index  $i$  such that*

$$\text{Adv}_{\Phi,t}^{\text{oup1}}(\mathcal{A}) \leq \text{Adv}_{\Phi,t}^{\text{oup}}(\mathcal{B}_1) + \frac{\ell(1 + q_\pi)}{2^n - \ell} .$$

*Finally, for any adversary  $\mathcal{A}$  against CF1 with parameters as before, there are adversaries  $\mathcal{B}_1$  and  $\mathcal{B}_2$ , where  $\mathcal{B}_1$  is as in the previous case, and  $\mathcal{B}_2$  outputs a list of size  $\ell$  and makes  $q_i$  permutation queries with index  $i$  such that*

$$\text{Adv}_{\Phi,t}^{\text{cf1}}(\mathcal{A}) \leq \text{Adv}_{\Phi,t}^{\text{oup}}(\mathcal{B}_1) + 2 \cdot \text{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{B}_2) + \frac{\ell}{2^n - \ell} + \frac{\ell}{2^n - 2\ell} .$$

For the first inequality, given  $\mathcal{A}$  against OUP outputting List of size  $\ell$ , algorithm  $\mathcal{B}_1$  against OUP1 runs  $\mathcal{A}$ , simulates its  $\pi$  queries using its own  $\pi$  oracle, and constructs a new list List' consisting of tuples  $(1, +, \phi, 0, P_1(c) \oplus c)$  for each  $(\phi, c) \in \text{List}$ . Now if List contains an entry  $(\phi, c)$  such that  $\phi(k) = c$ , then the corresponding entry  $(1, +, \phi, 0, c')$  on List' would satisfy  $P_1(\phi(k) \oplus 0) \oplus \phi(k) = c'$ . Note that List' is also of size  $\ell$ , but  $\mathcal{B}_1$  places  $\ell$  extra queries to  $P_1$ .

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

For the second inequality, given  $\mathcal{A}$ 's output  $\text{List}$  of size  $\ell$ , algorithm  $\mathcal{B}_2$  runs  $\mathcal{A}$ , simulates its  $\pi$  queries using its own  $\pi$  oracle, and constructs a new list  $\text{List}'$  consisting of tuples of the form  $(1, +, \phi_1, 0, \phi_2, 0)$  for each  $(\phi_1, \phi_2) \in \text{List}$ . If  $\text{List}$  contains an entry  $(\phi_1, \phi_2)$  such that  $\phi_1(k) = \phi_2(k)$ , then the corresponding entry  $(1, +, \phi_1, 0, \phi_2, 0)$  on  $\text{List}'$  would satisfy  $\text{P}_1(\phi_1(k) \oplus 0) \oplus \phi_1(k) = \text{P}_1(\phi_2(k) \oplus 0) \oplus \phi_2(k)$ . The size of  $\text{List}'$  is also  $\ell$  and  $\mathcal{B}_2$  places the same number of queries to  $\text{P}_i^\pm$  as  $\mathcal{A}$ .

For the the third inequality, let us consider a modified OUP1 game where the  $\pi$  oracle used in the winning condition is replaced with an independent random permutation  $\pi'$ . Since the outputs of  $\pi'$  are independent of  $\mathcal{A}$ 's view, each entry in  $\mathcal{A}$ 's list wins the game with probability at most  $1/(2^n - \ell + 1)$ , and hence  $\mathcal{A}$ 's advantage is at most  $\ell/(2^n - \ell)$ . Furthermore, these two games are identical unless  $\mathcal{A}$ 's list of  $\pi$  queries  $(i, a, b)$ , where either  $b = \pi(i, a, +)$  or  $a = \pi(i, b, -)$ , together with  $\pi'$  list of queries  $(i, a', b')$ , where  $b' = \pi(i, a', +)$  or  $a' = \pi(i, b', -)$  contradict the required permutivity of  $\pi(i, \cdot, \cdot)$ . For this it is sufficient to bound the probability that there are two entries with matching first or second entries. There are two possibilities.

Firstly, an input to  $\pi'$  may match an input or output to  $\pi$ . We can reduce this to output unpredictability. Given  $\mathcal{A}$ , algorithm  $\mathcal{B}_1$  runs  $\mathcal{A}$  and handles its  $\pi$  queries using its own  $\pi$  oracle. Note that this simulation is perfect. Algorithm  $\mathcal{B}_1$  keeps track of  $\mathcal{A}$ 's queries to  $\pi$  via  $\text{List}_\pi$  containing entries  $(i, a, b)$ . When  $\mathcal{A}$  outputs  $\text{List}$  containing entries  $(i, \sigma, \phi, x, c)$ , algorithm  $\mathcal{B}_1$  returns a list containing  $(\phi, a \oplus x)$  and  $(\phi, b \oplus x)$  for  $(i, *, \phi, x, *) \in \text{List}$  and  $(i, a, *), (i, *, b) \in \text{List}_\pi$  for all  $i$ . Thus  $\mathcal{B}_1$  wins the OUP game with a list of size at most  $2 \sum_i \ell_i q_i$ , where  $\ell_i$  is the size of entries in  $\text{List}$  with index  $i$ . (With more careful counting the factor 2 can be avoided.)

Secondly, an output of  $\pi'$  may match an input or output to  $\pi$ . Here we cannot reduce to output unpredictability as  $\pi'$  is not available to  $\mathcal{B}_1$ . But this is not needed:  $\pi'$  is independent of  $\mathcal{A}$ 's view and the probability of this event is bounded by  $\sum_i (\ell_i q_i) / (2^n - \min(\ell_i, q_i))$ .

The inequality follows by noting that  $\sum_i (\ell_i q_i) \leq \ell q_\pi$  and  $\max(\ell, \min(\ell_i, q_i)) \leq \ell$ .

To prove the final inequality, again we consider a modified game where the winning condition is performed with respect to an independent permutation  $\pi'$ . The

#### 4.4 The RKCPA security of $\text{EM}^\pi[1, 1, 1]$

---

change in  $\mathcal{A}$ 's success probability can be bounded as in the previous case down to output unpredictability. We modify this game further, by considering a game whose winning requirement is changed to that of the CF game: given a list of entries  $(i, \sigma, \phi_1, x_1, \phi_2, x_2)$  check if  $\phi_1(k) = \phi_2(k)$  for some entry on the list. The outputs of these two games are identical unless one of the following takes place. (1) The second game outputs **false** and the third outputs **true**. In this, case we can construct an adversary which wins the CF game which simply outputs all pairs  $(\phi_1, \phi_2)$  in  $\mathcal{A}$ 's list. (2) The second game outputs **true** and the third outputs **false**. In this case, there are two sub-possibilities: (2.1) The adversary wins with a pair  $(i, \sigma, \phi_1, x_1, \phi_2, x_2)$  such that  $\phi_1(x_1) \oplus x_1 = \phi_2(x_2) \oplus x_2$  (but of course  $\phi_1(k) \neq \phi_2(k)$ ). This cannot be the case as  $\pi'$  is a permutation. (2.2) Adversary  $\mathcal{A}$  wins with a pair  $(i, \sigma, \phi_1, x_1, \phi_2, x_2)$  such that  $\phi_1(x_1) \oplus x_1 \neq \phi_2(x_2) \oplus x_2$ . As before since  $\pi'$  is independent of  $\mathcal{A}$ 's view, the probability of this event is at most  $\ell/(2^n - 2\ell)$ , since each entry places 2 queries to  $\pi$ . Finally note that the final game is identical to the CF game (and oracle  $\pi'$  is not used by the game).

Bellare and Kohno [27] show that the RKD set  $\Phi^\oplus$  is output unpredictable with advantage  $\ell/2^n$  for any adversary outputting a list of size  $\ell$ , and claw-free with advantage 0. The above proposition allow us to conclude that this set is also first-order output unpredictable and first-order claw-free.

**Corollary 4.4.** *Let  $t \geq 1$  and suppose  $\Phi^\oplus$  is defined with respect to a key space of size  $2^n$ . Then for any  $\mathcal{A}$  outputting a list of at most  $\ell \leq 2^n/4$  and making at most  $q_1$  queries to its  $\mathsf{P}_1$  oracle,*

$$\text{Adv}_{\Phi^\oplus, t}^{\text{oup1}}(\mathcal{A}) \leq \frac{\ell(q_1 + 1)}{2^{n-1}} \quad \text{and} \quad \text{Adv}_{\Phi^\oplus, t}^{\text{cf1}}(\mathcal{A}) \leq \frac{\ell(q_1 + 2)}{2^{n-1}} .$$

This corollary, together with Theorem 4.2, allows us to establish that  $\text{EM}^\pi[1, 1, 1]$  is  $\Phi^\oplus$ -RKCPA secure.

**Corollary 4.5.** *For an adversary  $\mathcal{A}$  against the  $\Phi^\oplus$ -RKCPA security of  $\text{EM}^\pi[1, 1, 1]$  that makes at most  $q_\pi$  queries to its  $\pi$  oracle (of which  $q_i$  are to  $\pi(i, \cdot, \cdot)$ ) and at most  $q_{em}$  queries to its RK-ENC oracle, with  $q_2, q_{em} < 2^n/2$*

$$\text{Adv}_{\text{EM}^\pi[1,1,1], \Phi^\oplus, 2}^{\text{rkcpa}} \leq \frac{2q_{em}(q_2 + q_{em})(q_1 + 3)}{2^n} + \frac{q_2 q_{em}}{2^n - q_2} .$$

Via a direct analysis (but at the expense of modularity) the cubic bound above can be tightened to a quadratic one.

#### 4.4 The RKCPA security of $EM^\pi[1, 1, 1]$

---

##### 4.4.4 A $\Phi^\oplus$ -RKCCA attack on $EM^\pi[1, 1, 1]$

The above result raises the question of whether the security proof can be extended to the CCA setting. Adapting an attack due to Andreeva et al. [9] on the indistinguishability of the two-round EM construction to the RKA setting, we show that  $EM^\pi[1, 1, 1]$  is not  $\Phi^\oplus$ -RKCCA secure. The adversary is shown below, where  $\bar{x}$  denotes  $x \oplus 1^n$ , and (slightly abusing notation)  $\Delta \in \{0, 1\}^n$  denotes the function  $k \mapsto k \oplus \Delta$ . RK-ENC and RK-DEC are as defined in Section 2.3.2.

$\mathcal{A}^{\text{RK-ENC, RK-DEC, } \pi}$ :

Query RK-ENC( $0^n, 0^n$ ); Get  $y_0$

Query RK-ENC( $1^n, 1^n$ ); Get  $y_1$

Query RK-DEC( $1^n, \bar{y}_0$ ); Get  $x$

Query RK-ENC( $0^n, \bar{x}$ ); Get  $y_2$

Return ( $y_2 = \bar{y}_1$ )

When interacting with oracles implementing the EM construction, we show that  $\mathcal{A}$  returns true with probability 1. We have that  $y_0 = P_2(P_1(k) \oplus k) \oplus k$  and  $y_1 = P_2(P_1(k) \oplus \bar{k}) \oplus \bar{k}$ . Now  $x$  is calculated as

$$\begin{aligned}
 \bar{y}_0 &= P_2(P_1(k) \oplus k) \oplus k \oplus 1^n \xrightarrow{\oplus \bar{k}} P_2(P_1(k) \oplus k) \\
 &\xrightarrow{P_2^{-1}} P_1(k) \oplus k \\
 &\xrightarrow{\oplus \bar{k}} \overline{P_1(k)} \\
 &\xrightarrow{P_1^{-1}} P_1^{-1}(\overline{P_1(k)}) \\
 &\xrightarrow{\oplus \bar{k}} P_1^{-1}(\overline{P_1(k)}) \oplus \bar{k} = x .
 \end{aligned}$$

Variable  $y_2$  is calculated as

$$\begin{aligned}
 \bar{x} &= P_1^{-1}(\overline{P_1(k)}) \oplus \bar{k} \oplus 1^n \xrightarrow{\oplus k} P_1^{-1}(\overline{P_1(k)}) \\
 &\xrightarrow{P_1} \overline{P_1(k)} \\
 &\xrightarrow{\oplus k} \overline{P_1(k)} \oplus k \\
 &\xrightarrow{P_2} P_2(\overline{P_1(k)} \oplus k) \\
 &\xrightarrow{\oplus k} P_2(\overline{P_1(k)} \oplus k) \oplus k .
 \end{aligned}$$

## 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

---

Hence  $y_2 = P_2(P_1(k) \oplus \bar{k}) \oplus k = \bar{y}_1$ . On the other hand, when the adversary is interacting with the ideal cipher, for the equality to hold we need to have that

$$E_k(D_{\bar{k}}(\overline{E_k(0)})) = \overline{E_{\bar{k}}(1)} \quad \text{i.e.,} \quad D_{\bar{k}}(\overline{E_k(0)}) = D_k(\overline{E_{\bar{k}}(1)}).$$

The latter equality however holds with negligible probability. This attack also applies if the round permutations are identical, i.e., when  $P_2 = P_1$ .

Note that in the CCA setting we would need to simulate both permutations  $P_1$  and  $P_2$  forgetfully as forward and backward outputs need to look random. To do this we would have to re-introduce the xor claw-free condition in order to rule out collisions on  $P_1$ , which in turn excludes the  $\Phi^\oplus$  set.

It is instructive to check where the above sequence of queries triggers collisions in the second permutation, irrespectively of how  $P_1$  is simulated. Let  $z = P_1(k) \oplus k$ . During the first and second RK-ENC queries,  $P_2$  is queried on points  $z$  and  $\bar{z}$ , respectively. During the decryption query,  $P_2^{-1}$  is queried on  $P_2(z)$ , which is equivalent to  $P_2$  being queried on  $z$ . This is a  $P_2$  collision. Note also that in the third RK-ENC query a second collision occurs as  $P_2$  is queried on  $\bar{z}$ .

## 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

Building on the results of the previous sections, we set out to find a key schedule for the iterated Even–Mansour construction that provides  $\Phi^\oplus$ -RKCCA security. Our previous results show that at least three rounds are necessary. We start by showing that of the fourteen possible simple key schedules for three-round EM, all but one fall prey to  $\Phi^\oplus$ -RKCCA attacks. We then show that the remaining  $\text{EM}^\pi[1, 1, 1, 1]$  construction does indeed provide  $\Phi^\oplus$ -RKCCA security.

### 4.5.1 Attacking $\text{EM}^\pi[\kappa]$ for $\kappa \neq [1, 1, 1, 1]$

Up to relabelling, there are 14 possible key schedules for the three-round Even–Mansour schemes. Of these, 9 are susceptible to offset-switching attacks; these are key schedules where a key appears only in the first or last rounds and nowhere

#### 4.5 The RKCCA security of $EM^\pi[1, 1, 1, 1]$

$\mathcal{A}^{\text{RK-ENC,RK-DEC},\pi}$ : Query RK-ENC(00, $0^n$ );    Get $y_0$ Query RK-ENC(10, $1^n$ );    Get $y_1$ Query RK-DEC(10, $\overline{y_0}$ );    Get $x$ Query RK-ENC(00, $\overline{x}$ );    Get $y_2$ Return ( $y_2 = \overline{y_1}$ )	$\mathcal{A}^{\text{RK-ENC,RK-DEC},\pi}$ : Query RK-ENC(00, $0^n$ );    Get $y_0$ Query RK-ENC(10, $1^n$ );    Get $y_1$ Query RK-DEC(10, $y_0$ );    Get $x$ Query RK-ENC(00, $\overline{x}$ );    Get $y_2$ Return ( $y_2 = y_1$ )
$\mathcal{A}^{\text{RK-ENC,RK-DEC},\pi}$ : Query RK-ENC(00, $0^n$ );    Get $y_0$ Query RK-ENC(10, $1^n$ );    Get $y_1$ Query RK-DEC(10, $y_0$ );    Get $x$ Query RK-ENC(00, $\overline{x}$ );    Get $y_2$ Return ( $y_2 = y_1$ )	

Figure 4.8: Adversaries attacking  $EM^\pi[1, 1, 2, 1]$  (top left),  $EM^\pi[1, 1, 2, 2]$  (top right), and  $EM^\pi[1, 2, 1, 2]$  (bottom). Here,  $c_0c_1 \in \{0, 1\}^2$  denotes the RKD function  $(k_1, k_2) \mapsto (k_1 \oplus c_1^n, k_2 \oplus c_2^n)$ .

else such as  $[1, 2, 2, 2]$ ,  $[1, 2, 2, 3]$ , or  $[1, 2, 2, 1]$  and this rules out 9 key schedules. Another 4 can be attacked using Andreeva et al.’s attack [9]; these are  $[1, 1, 2, 1]$ ,  $[1, 2, 1, 1]$ ,  $[1, 1, 2, 2]$ , and  $[1, 2, 1, 2]$  schedules. We give three attacks in Figure 4.8 where  $c_0c_1 \in \{0, 1\}^2$  denotes the RKD function  $(k_1, k_2) \mapsto (k_1 \oplus c_1^n, k_2 \oplus c_2^n)$ .

The analysis of the success probabilities of these adversaries are similar to that for the attack in Section 4.4.4 and hence is omitted.

These attacks give a generic 4-query related-key distinguisher for reduced-round LED (8 out of 32 rounds for LED-64 and 16 out of 48 for LED-128). However, our results in the following section support the designers’ claim that LED-64 provides good related-key attack security despite the simple key schedule.

#### 4.5.2 The security of $EM^\pi[1, 1, 1, 1]$

We now study the only remaining construction,  $EM^\pi[1, 1, 1, 1]$ ; we identify conditions under which it provides  $\Phi$ -RKCCA security, examine its  $\Phi^\oplus$ -RKA security, and briefly discuss the impact of permutation reuse on this construction.

## 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

---

### 4.5.2.1 Sufficient conditions for RKCCA security

We now show that  $\text{EM}^\pi[1, 1, 1, 1]$ , achieves  $\Phi$ -RKCCA security for sets  $\Phi$  satisfying the conditions described in Sections 4.3.1 and 4.4.1. As before we motivate a number of restrictions on  $\Phi$  by considering a simulation strategy and analysing the inconsistencies that could arise. The adversary in the  $\Phi$ -RKCCA game with respect to the construction has access to  $\pi$  and the oracles

$$\begin{aligned} & \mathsf{P}_3(\mathsf{P}_2(\mathsf{P}_1(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^{\mathsf{P}_i}(k) , \\ & \mathsf{P}_1^{-1}(\mathsf{P}_2^{-1}(\mathsf{P}_3^{-1}(y \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) . \end{aligned}$$

Once again we aim to simulate the above two oracles by returning uniformly random values. There are at least two way to perform this:

- (a) Simulate the outer permutations (i.e. the  $\mathsf{P}_3$  oracle in RK-ENC and the  $\mathsf{P}_1^{-1}$  oracle in RK-DEC) forgetfully.
- (b) Simulate the middle oracles  $\mathsf{P}_2$  and  $\mathsf{P}_2^{-1}$  forgetfully. This ensures that the inputs to the  $\mathsf{P}_1^\pm$  and  $\mathsf{P}_3^\pm$  are randomised, so their outputs are also random.

The first approach, in some sense the more natural one, does not work. This is because  $\mathsf{P}_1$  (respectively  $\mathsf{P}_3$ ) also appear as the first-round permutation in RK-ENC (respectively RK-DEC). An adversary which performs an offset switch can trigger collisions in these oracles without being detected. We therefore adopt the second simulation strategy and, for a forgetful oracle  $\$$ , consider

$$\begin{aligned} & \mathsf{P}_3(\$ (\mathsf{P}_1(x \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) , \\ & \mathsf{P}_1^{-1}(\$ (\mathsf{P}_3^{-1}(y \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) . \end{aligned}$$

We now consider inconsistencies, starting with a query collision between  $\pi$  (from a query of  $\mathcal{A}$ ) and  $\$$  arising from either the forward or backwards direction. Here we rely on first-order output unpredictability, but note that  $(i, \sigma) = (1, +)$  and  $(i, \sigma) = (3, -)$  will be critically relied on. Collisions arising between an RKD query to  $\pi$  and a  $\$$  query in either direction can be ruled out down to first-order query independence; once again  $(i, \sigma) \in \{(1, +), (3, -)\}$  will be used. Finally, the probability

#### 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

---

that a collision occurs as a result of two queries to  $\$$  and/or  $\$^{-1}$  can be bounded by the first-order claw freeness property. As before, inconsistencies also arise due to collisions between the outputs of oracle queries; the probability of this occurring can be bounded information-theoretically. Note that here we also rely on independence of queries to the second permutation, but both cases  $(i, \sigma) \in \{(1, +), (3, -)\}$  in the definition will be relied on. We now formally prove the following theorem.

**Theorem 4.6** ( $\Phi$ -RKCCA security of  $\text{EM}^\pi[1, 1, 1, 1]$ ). *Let  $\Phi$  be an RKD set. Then for any adversary  $\mathcal{A}$  against the  $\Phi$ -RKCCA security of  $\text{EM}^\pi[1, 1, 1, 1]$  with parameters as before, we have adversaries  $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$ , and  $\mathcal{B}_4$  such that*

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1,1],\Phi,3}^{\text{rkcca}} &\leq \text{Adv}_{\Phi,3}^{\text{oup}1}(\mathcal{B}_1) + \text{Adv}_{\Phi,3}^{\text{qi}1}(\mathcal{B}_2) \\ &\quad + 2\text{Adv}_{\Phi,3}^{\text{cf}1}(\mathcal{B}_3) + \text{Adv}_{\Phi,3}^{\text{cf}}(\mathcal{B}_4) \\ &\quad + \frac{2q_{em}^2}{2^n} + \frac{2q_{em}(q_2 + \sum_{\phi} q_2^{\phi})}{2^n - (q_2 + \sum_{\phi} q_2^{\phi})}, \end{aligned}$$

where  $\mathcal{B}_1$  outputs a list of length  $2q_2q_{em}$ ,  $\mathcal{B}_2$  a list of length  $2q_{em}^2$ ,  $\mathcal{B}_3$  a list of length  $q_{em}^2$ , and  $\mathcal{B}_4$  a list of length at most  $q_{em}^2$ .

The proof follows a similar pattern to the proof of Theorems 4.1 and 4.2; again we proceed through four stages. In the first,  $\mathcal{A}$  interacts with the public permutations and their inverses, plus the 3-round Even–Mansour scheme (and its inverse, omitted here for conciseness) instantiated with the same permutations:

$$\pi(i, x, \sigma), \quad \text{P}_3(\text{P}_2(\text{P}_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k).$$

We then consider an environment in which  $\text{P}_2$  is replaced by  $\$$ , a forgetful random oracle, for queries made to the Even–Mansour scheme:

$$\pi(i, x, \sigma), \quad \text{P}_3(\$(\text{P}_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)$$

and from here we consider a keyed random function:

$$\pi(i, x, \sigma), \quad \text{iF}(\phi^\pi(k), x).$$

Finally, we transition to a game in which  $\text{iF}$  is replaced by an ideal cipher ( $\text{iE}, \text{iD}$ ):

$$\pi(i, x, \sigma), \quad \text{iE}(\phi^\pi(k), x).$$

#### 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

<p><u>Game <math>i</math>:</u>  <math>k \leftarrow_{\\$} \mathcal{K}</math>  <math>b' \leftarrow_{\\$} \mathcal{A}^{\text{RK-ENC, RK-DEC}, \pi}</math>            Return <math>b'</math></p> <p><u>RK-ENC(<math>\phi^\pi, x</math>):</u>  <math>k' \leftarrow \phi^\pi(k)</math>  <math>z_1 \leftarrow \mathbf{S}_1(k' \oplus x)</math>  <math>z_2 \leftarrow \mathbf{IS}_2(k' \oplus z_1)</math>            Return <math>k' \oplus \mathbf{S}_3(k' \oplus z_2)</math></p> <p><u><math>\pi(1, a, +)</math>:</u>            Return <math>\mathbf{S}_1(a)</math></p> <p><u><math>\pi(2, a, +)</math>:</u>            Return <math>\mathbf{DS}_2(a)</math></p> <p><u><math>\pi(3, a, +)</math>:</u>            Return <math>\mathbf{S}_3(a)</math></p>	<p><u><math>\mathbf{S}_1(a)</math>:</u>            If <math>\mathbf{S}_1[a] \neq \perp</math>                Return <math>\mathbf{S}_1[a]</math>  <math>b \leftarrow_{\\$} \{0, 1\}^n \setminus \text{Rng}(\mathbf{S}_1)</math>  <math>\mathbf{S}_1[a] \leftarrow b; \mathbf{S}_1^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(\mathbf{S}_1) \leftarrow \text{Rng}(\mathbf{S}_1) \cup \{b\}</math>  <math>\text{Dom}(\mathbf{S}_1) \leftarrow \text{Dom}(\mathbf{S}_1) \cup \{a\}</math>            Return <math>\mathbf{S}_1[a]</math></p> <p><u><math>\mathbf{S}_3(a)</math>:</u>            If <math>\mathbf{S}_3[a] \neq \perp</math>                Return <math>\mathbf{S}_3[a]</math>  <math>b \leftarrow_{\\$} \{0, 1\}^n \setminus \text{Rng}(\mathbf{S}_3)</math>  <math>\mathbf{S}_3[a] \leftarrow b; \mathbf{S}_3^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(\mathbf{S}_3) \leftarrow \text{Rng}(\mathbf{S}_3) \cup \{b\}</math>  <math>\text{Dom}(\mathbf{S}_3) \leftarrow \text{Dom}(\mathbf{S}_3) \cup \{a\}</math>            Return <math>\mathbf{S}_3[a]</math></p>
---	---

Figure 4.9: Procedures common to all games in the proof of Theorem 4.6. Oracles  $\text{RK-DEC}$ ,  $\pi(i, \cdot, -)$ ,  $\mathbf{S}_i^{-1}$ ,  $\mathbf{DS}_2^{-1}$ , and  $\mathbf{IS}_2^{-1}$  are defined in a similar way to their corresponding forward oracles.

We will now argue that the above changes alter  $\mathcal{A}$ 's winning probabilities negligibly and bound  $\mathcal{A}$ 's winning probability in terms of the conditions on  $\Phi$  introduced in Sections 4.3.1 and 4.4.1.

The first transition is analysed via a series of games, given in Figures 4.9, 4.10, and 4.11. These games include two intermediate transitions: in the first,  $\mathbf{P}_2$  is replaced with  $\mathbf{Q}$  (a random permutation, chosen independently of  $\pi$ ) for queries arising through  $\text{RK-ENC}$  or  $\text{RK-DEC}$ ; in the second,  $\mathbf{Q}$  is replaced with  $\mathbf{\$}$  (a forgetful random oracle). We identify the points at which these two intermediate transitions lead to inconsistencies, by setting bad flags.

We omit a specification of the inverse oracles for conciseness; they are defined analogously to their respective forward oracles. As before, the event where the adversary outputs 1 in game  $i$  is denoted by  $S_i$ .

#### 4.5 The RKCCA security of $EM^\pi[1, 1, 1, 1]$

<p><u>Game 1:</u></p> <p><u>DS<sub>2</sub>(a):</u>          If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math>          If <math>I_2[a] \neq \perp</math>              Return <math>I_2[a]</math>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2)</math>          If <math>b \in \text{Rng}(IS_2)</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math>  <math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_2) \leftarrow \text{Rng}(DS_2) \cup \{b\}</math>  <math>\text{Dom}(DS_2) \leftarrow \text{Dom}(DS_2) \cup \{a\}</math>          Return <math>D_2[a]</math></p> <p><u>IS<sub>2</sub>(a):</u>          If <math>I_2[a] \neq \perp</math>              Return <math>I_2[a]</math>          If <math>D_2[a] \neq \perp</math>              Return <math>D_2[a]</math>  <math>b \leftarrow_s \{0, 1\}^n</math>          If <math>b \in \text{Rng}(IS_2)</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(IS_2)</math>          If <math>b \in \text{Rng}(DS_2)</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math>  <math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_2) \leftarrow \text{Rng}(IS_2) \cup \{b\}</math>  <math>\text{Dom}(IS_2) \leftarrow \text{Dom}(IS_2) \cup \{a\}</math>          Return <math>I_2[a]</math></p>	<p><u>Game 2</u> <u>Game 3:</u></p> <p><u>DS<sub>2</sub>(a):</u>          If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math>          If <math>I_2[a] \neq \perp</math>              <math>\text{bad}_1 \leftarrow \text{true};</math> <span style="border: 1px solid black; padding: 2px;">Return <math>I_2[a]</math></span>  <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2)</math>          If <math>b \in \text{Rng}(IS_2)</math>              <math>\text{bad}_2 \leftarrow \text{true}</math>              <span style="border: 1px solid black; padding: 2px;"><math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math></span>  <math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(DS_2) \leftarrow \text{Rng}(DS_2) \cup \{b\}</math>  <math>\text{Dom}(DS_2) \leftarrow \text{Dom}(DS_2) \cup \{a\}</math>          Return <math>D_2[a]</math></p> <p><u>IS<sub>2</sub>(a):</u>          If <math>I_2[a] \neq \perp</math>              Return <math>I_2[a]</math>          If <math>D_2[a] \neq \perp</math>              <math>\text{bad}_1 \leftarrow \text{true};</math> <span style="border: 1px solid black; padding: 2px;">Return <math>D_2[a]</math></span>  <math>b \leftarrow_s \{0, 1\}^n</math>          If <math>b \in \text{Rng}(IS_2)</math>                <math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(IS_2)</math>          If <math>b \in \text{Rng}(DS_2)</math>              <math>\text{bad}_2 \leftarrow \text{true}</math>              <span style="border: 1px solid black; padding: 2px;"><math>b \leftarrow_s \{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)</math></span>  <math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math>  <math>\text{Rng}(IS_2) \leftarrow \text{Rng}(IS_2) \cup \{b\}</math>  <math>\text{Dom}(IS_2) \leftarrow \text{Dom}(IS_2) \cup \{a\}</math>          Return <math>I_2[a]</math></p>
--	--

Figure 4.10: Games 1 to 3 in the proof of Theorem 4.6. Oracles  $\pi(i, \cdot, -)$ ,  $S_i^{-1}$ ,  $DS_2^{-1}$ , and  $IS_2^{-1}$  are defined in a similar way to their corresponding forward oracles. Boxed statements are included in Game 2 and are omitted from Game 3.

#### 4.5 The RKCCA security of $EM^\pi[1, 1, 1, 1]$

---

Game 4	Game 5:
<p><u><math>DS_2(a)</math>:</u></p> <p>If <math>D_2[a] \neq \perp</math> Return <math>D_2[a]</math></p> <p>If <math>I_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_1 \leftarrow true</math></p> <p><math>b \leftarrow_s \{0, 1\}^n \setminus Rng(DS_2)</math></p> <p>If <math>b \in Rng(IS_2)</math></p> <p style="padding-left: 20px;"><math>bad_2 \leftarrow true</math></p> <p><math>D_2[a] \leftarrow b; D_2^{-1}[b] \leftarrow a</math></p> <p><math>Rng(DS_2) \leftarrow Rng(DS_2) \cup \{b\}</math></p> <p><math>Dom(DS_2) \leftarrow Dom(DS_2) \cup \{a\}</math></p> <p>Return <math>D_2[a]</math></p> <p><u><math>IS_2(a)</math>:</u></p> <p>If <math>I_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_3 \leftarrow true; \boxed{\text{Return } I_2[a]}</math></p> <p>If <math>D_2[a] \neq \perp</math></p> <p style="padding-left: 20px;"><math>bad_1 \leftarrow true</math></p> <p><math>b \leftarrow_s \{0, 1\}^n</math></p> <p>If <math>b \in Rng(IS_2)</math></p> <p style="padding-left: 20px;"><math>bad_4 \leftarrow true</math></p> <p style="padding-left: 40px;"><math>\boxed{b \leftarrow_s \{0, 1\}^n \setminus Rng(IS_2)}</math></p> <p>If <math>b \in Rng(DS_2)</math></p> <p style="padding-left: 20px;"><math>bad_2 \leftarrow true</math></p> <p><math>I_2[a] \leftarrow b; I_2^{-1}[b] \leftarrow a</math></p> <p><math>Rng(IS_2) \leftarrow Rng(IS_2) \cup \{b\}</math></p> <p><math>Dom(IS_2) \leftarrow Dom(IS_2) \cup \{a\}</math></p> <p>Return <math>I_2[a]</math></p>	

Figure 4.11: Games 4 and 5 in the proof of Theorem 4.6. Oracles  $\pi(i, \cdot, -)$ ,  $S_i^{-1}$ ,  $DS_2^{-1}$ , and  $IS_2^{-1}$  are defined in a similar way to their corresponding forward oracles. Boxed statements are included in Game 4 and are omitted from Game 5.

#### 4.5 The RKCCA security of $EM^\pi[1, 1, 1, 1]$

---

**Game 0** is the RKA game augmented with a public permutation oracle (as described in Section 2.3.2), conditioned on  $b = 1$ . In this game, the adversary interacts with an oracle realising the public permutations  $\pi$  and the Even–Mansour construction instantiated with  $\pi$ .

**Game 1** is only syntactically different from **Game 0**. Sampling algorithms  $S_1$  and  $S_3$  are introduced to respond to queries made to  $\pi(1, \cdot, +)$  and  $\pi(3, \cdot, +)$ . Queries to  $\pi(2, \cdot, +)$  are split into two groups: those made directly to  $\pi$ , either by the adversary or by an RKD function, which are answered by the sampling algorithm  $DS_2$  and those made indirectly (through queries made to RK-ENC), which are answered by  $IS_2$ . Appropriate inverse sampling algorithms are also introduced. The oracles  $DS_2$  and  $IS_2$  maintain consistent lists,  $D_2$  and  $I_2$ . The lists used by inverse oracles are identical to the lists used by the corresponding forward oracles. As this is a purely syntactic change,  $\Pr[S_0] = \Pr[S_1]$ .

**Game 2** sets  $\text{bad}_1$  either if  $DS_2$  is queried on a point already defined in  $I_2$  or if  $IS_2$  is queried on a point already defined in  $D_2$  (and similarly for the inverse oracles). This occurs either because  $\mathcal{A}$  queries  $\pi(2, \cdot, \cdot)$  directly at a point that is also queried to  $\pi(2, \cdot, \cdot)$  through an indirect RK-ENC query, or because an RKD function queries  $\pi(2, \cdot, \cdot)$  at a point that is also queried to  $\pi(2, \cdot, \cdot)$  through an RK-ENC query (and similarly for the inverse oracles). We will later bound the probability of this event in terms of the first-order output unpredictability and first-order query independence of  $\Phi$ . **Game 2** sets  $\text{bad}_2$  if the value chosen at random for  $IS_2(a)$  is already defined in range of  $DS_2$ , or vice versa (and similarly for the inverse queries and the domain of  $IS_2$  or  $DS_2$ ). This is necessary because in **Game 1**, for both  $DS_2$  and  $IS_2$ ,  $b$  is sampled from  $\{0, 1\}^n \setminus \text{Rng}(DS_2, IS_2)$  whereas our objective in **Game 3** is to ensure that responses from  $DS_2$  are independent of responses from  $IS_2$ . The code of  $S_1$  and  $S_3$  remains unchanged throughout this proof. The outputs of  $DS_2$  and  $IS_2$  remain consistent and  $\Pr[S_1] = \Pr[S_2]$ .

**Game 3** omits the boxed statements in **Game 2** and so is identical to **Game 2** unless one of  $\text{bad}_1$  or  $\text{bad}_2$  is set. In this game, the oracles  $DS_2$  and  $IS_2$  check consistency with their own lists, but may become inconsistent with each other. It is possible for  $\text{bad}_1$  to be set in two possible ways: event  $E_1$  is the event an adversary directly queries  $DS_2$  at a point coinciding with a point queried to  $IS_2$  from a query to RK-ENC (or comparable conditions resulting from queries to

#### 4.5 The RKCCA security of $EM^\pi[1, 1, 1, 1]$

---

inverse oracles); event  $E_2$  is the event an RKD function queries  $DS_2$  at a point coinciding with a point queried to  $IS_2$  from a query to RK-ENC (or comparable conditions resulting from queries to inverse oracles). We will analyse each of the ways that  $\text{bad}_1$  can be set below. Similarly,  $\text{bad}_2$  can be set either because of a query to  $DS_2$  from  $\mathcal{A}$ , a query to  $DS_2$  from  $\phi^\pi$ , or from a query to  $IS_2$  due to a query to RK-ENC (or similarly for the corresponding inverse oracles); we consider all cases simultaneously below. In Game 3, the responses to RK-ENC queries are completely decoupled from the responses to  $\pi$  queries, so we can consider that RK-ENC uses  $\mathbf{Q}$  to respond to queries and  $\pi$  uses  $\mathbf{P}_2$ . We have that  $\Pr[S_2] \leq \Pr[S_3] + \Pr[E_1 \vee E_2 \vee \text{bad}_2]$ .

**Game 4** sets  $\text{bad}_3$  if two distinct queries to RK-ENC result in the same value being queried to  $IS_2$ . The flag  $\text{bad}_3$  can be set on a query to RK-ENC or RK-DEC; these cases are dealt considered when we analyse the probability of setting bad events below. Game 4 chooses the response to  $IS_2$  uniformly from  $\{0, 1\}^n$  and sets  $\text{bad}_4$  if this value is already in  $\text{Rng}(IS_2)$ . Game 4 is equivalent to Game 3 and, in particular,  $\Pr[S_3] = \Pr[S_4]$ .

**Game 5** omits the boxed statements from Game 4 and so is identical to Game 4 unless  $\text{bad}_3$  or  $\text{bad}_4$  is set. Let  $E'_1, E'_2, \text{bad}'_2$  represent events in Game 5 corresponding to events  $E_1, E_2, \text{bad}_2$  in Game 4, then  $\Pr[E_1 \vee E_2 \vee \text{bad}_2] \leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2\Pr[\text{bad}_3 \vee \text{bad}_4]$  In this game, calls to  $\pi(2, \cdot, \cdot)$  through RK-ENC (RK-DEC), which are answered by  $IS_2$  ( $IS_2^{-1}$ ) are answered by a forgetful random oracle and so the ciphertexts (plaintexts) are uniform and independent of the key and the plaintext (ciphertexts).

In Game 5, the adversary interacts with

$$\pi(i, x, \sigma), \quad \mathbf{P}_3(\$(\mathbf{P}_1(\phi^\pi(k) \oplus x) \oplus \phi^\pi(k)) \oplus \phi^\pi(k)) \oplus \phi^\pi(k) .$$

During the next transition to

$$\pi(i, x, \sigma), \quad \text{iF}(\phi^{\mathbf{P}, \mathbf{P}^{-1}}(k), x) ,$$

inconsistencies only arise if the adversary makes queries  $(\phi_1^\pi, x_1) \neq (\phi_2^\pi, x_2)$ , but where  $(\phi_1^\pi(k), x_1) = (\phi_2^\pi(k), x_2)$ . If an adversary  $\mathcal{A}$  makes such a query, we can construct an adversary  $\mathcal{B}_4$  which wins the CF game with a list of length at most  $\frac{q_{em}^2}{2}$  as follows:  $\mathcal{B}_4$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(\phi_i^\pi, \phi_j^\pi) : 1 \leq i < j \leq q_{em}\}$ .

#### 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

---

Considering the final transition, we switch from a random function to a random permutation (for each  $\phi^\pi$ ); the probability of an inconsistency arising in this step is bounded by  $\frac{q_{em}^2}{2^n}$  [32].

Therefore we have that

$$\begin{aligned} \mathbf{Adv}_{\text{EM}^\pi[1,1,1,1],\Phi,t}^{\text{rkcca}}(\mathcal{A}) &\leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2\Pr[\text{bad}_3 \vee \text{bad}_4] \\ &\quad + \mathbf{Adv}_{\Phi,t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}. \end{aligned}$$

It remains to bound the probability that bad events occur in Game 5.

**Event  $E'_1$**  occurs when the adversary directly queries  $\pi(2, \cdot, \cdot)$  at a point that is also queried as a result of a query to RK-ENC. We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_1$  against the OUP1 game with a list of length  $2q_2q_{em}$ . The adversary  $\mathcal{B}_1$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, a_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_2\} \cup \{(3, -, \phi_i^\pi, y_i, b_j) : 1 \leq i \leq q_{em}, 1 \leq j \leq q_2\}$ . If  $\mathcal{A}$  sets  $\text{bad}_1$  with an RK-ENC or  $\text{DS}_2$  query, then  $\mathcal{B}_1$  wins the OUP1 game with a tuple of the form  $(1, +, \phi_i^\pi, x_i, a_j)$  and if  $\mathcal{A}$  sets  $\text{bad}_1$  with a query to RK-DEC or  $\text{DS}_2^{-1}$  then  $\mathcal{B}$  wins the OUP1 game with a tuple of the form  $(3, -, \phi_i^\pi, y_i, b_j)$ . We therefore conclude that  $\Pr[E'_1] \leq \mathbf{Adv}_{\Phi,t}^{\text{oup1}}(\mathcal{B}_1)$ , where  $\mathcal{B}_1$  outputs a list of length  $2q_2q_{em}$ .

**Event  $E'_2$**  occurs when an RKD function queries the  $\pi(2, \cdot, \cdot)$  at a point that is also queried as a result of a query to RK-ENC. We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_2$  against the QI1 game with a list of length  $2q_{em}^2$ . The adversary  $\mathcal{B}_2$  runs  $\mathcal{A}$  and outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, \phi_j^\pi) : 1 \leq i, j \leq q_{em}\} \cup \{(3, -, \phi_i^\pi, y_i, \phi_j^\pi) : 1 \leq i, j \leq q_{em}\}$ . If  $\mathcal{A}$  can set  $\text{bad}$  by causing an RKD function to query the permutation at a point that is also queried as a result of a query to RK-ENC, then the adversary  $\mathcal{B}_2$  will win the QI1 game with a tuple of a form  $(1, +, \phi_i^\pi, x_i, \phi_j^\pi)$  and if  $\mathcal{A}$  sets  $\text{bad}_1$  with a query to  $\text{IS}_2^{-1}$  or  $\pi(1, \cdot, -)$  then  $\mathcal{B}$  wins the QI1 game with a tuple of the form  $(3, -, \phi_i^\pi, y_i, \phi_j^\pi)$ . Therefore we can conclude that  $\Pr[E'_2] \leq \mathbf{Adv}_{\Phi,t}^{\text{qi1}}(\mathcal{B}_2)$ , where  $\mathcal{B}_2$  outputs a list of length  $2q_{em}^2$ .

**Flag  $\text{bad}'_2$**  is set with probability at most  $\frac{q_{em}(q_2 + \sum_\phi q_2^\phi)}{2^n - (q_2 + \sum_\phi q_2^\phi)}$ . It can be set in one of 16 different ways. Collisions between  $\text{DS}_2$  and  $\text{IS}_2$ ,  $\text{DS}_2^{-1}$  and  $\text{IS}_2^{-1}$ ,  $\text{DS}_2$

#### 4.5 The RKCCA security of $\text{EM}^\pi[1, 1, 1, 1]$

---

and  $\text{IS}_2^{-1}$ , or  $\text{DS}_2^{-1}$  and  $\text{IS}_2$  can all set  $\text{bad}_2$  and we count each case twice, depending on which query set  $\text{bad}_2$ . This gives 8 ways to set  $\text{bad}_2$ , however the query to  $\text{DS}_2$  can arise through a query by  $\mathcal{A}$  or through  $\phi^\pi$ , which gives 16 ways. In each case, we use a birthday-bound style argument, noting that each pair  $(x, a)$  has at most a  $\frac{1}{2^n - (q_1 + \sum_\phi q_1^\phi)}$  chance of setting  $\text{bad}'_2$ ; applying the union bound and recalling that  $q_{em}$  is the total number of queries made to RK-ENC and RK-DEC (and thus to  $\text{IS}$  and  $\text{IS}_2^{-1}$ ) by  $\mathcal{A}$  (and similarly for  $q_1$  and  $q_1^\phi$ ) gives the claimed probability.

**Flag**  $\text{bad}_3$  is set when two queries to RK-ENC result in the  $\text{P}_2$  being queried at the same point. We will use  $\mathcal{A}$  to create an adversary  $\mathcal{B}_3$  against the CF1 property of  $\Phi$ . The adversary  $\mathcal{B}_3$  runs  $\mathcal{A}$  and then outputs  $\text{List} = \{(1, +, \phi_i^\pi, x_i, \phi_j^\pi, x_j) : 1 \leq i < j \leq q_{em}\} \cup \{(3, -, \phi_i^\pi, y_i, \phi_j^\pi, y_j) : 1 \leq i < j \leq q_{em}\}$ . If  $\mathcal{A}$  sets  $\text{bad}_3$  with query to  $\text{IS}_2$ , then  $\mathcal{B}_3$  wins the CF1 game with a tuple of the form  $(1, +, \phi_i^\pi, x_i, \phi_j^\pi, x_j)$  and if  $\mathcal{A}$  sets  $\text{bad}_3$  with query to  $\text{IS}_2^{-1}$ , then  $\mathcal{B}_3$  wins the CF1 game with a tuple of the form  $(3, -, \phi_i^\pi, y_i, \phi_j^\pi, y_j)$ . Thus  $\Pr[\text{bad}_3] \leq \text{Adv}_{\Phi, t}^{\text{cf1}}(\mathcal{B}_3)$ , where  $\mathcal{B}_3$  outputs a list of length at most  $q_{em}^2$ .

**Flag**  $\text{bad}_4$  is set with probability at most  $\frac{q_{em}^2}{2} \frac{1}{2^n}$  using similar reasoning as in the setting of  $\text{bad}_2$ .

As we have that

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1,1], \Phi, t}^{\text{rkcca}}(\mathcal{A}) &\leq \Pr[E'_1 \vee E'_2 \vee \text{bad}'_2] + 2\Pr[\text{bad}_3 \vee \text{bad}_4] \\ &\quad + \text{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, \end{aligned}$$

we may conclude that

$$\begin{aligned} \text{Adv}_{\text{EM}^\pi[1,1,1,1], \Phi, t}^{\text{rkcca}} &\leq \text{Adv}_{\Phi, t}^{\text{oup1}}(\mathcal{B}_1) + \text{Adv}_{\Phi, t}^{\text{qi1}}(\mathcal{B}_2) + 2\frac{q_{em}(q_2 + \sum_\phi q_2^\phi)}{2^n - (q_2 + \sum_\phi q_2^\phi)} \\ &\quad + 2\left(\text{Adv}_{\Phi, t}^{\text{cf1}}(\mathcal{B}_3) + \frac{q_{em}^2}{2^{n+1}}\right) + \text{Adv}_{\Phi, t}^{\text{cf}}(\mathcal{B}_4) + \frac{q_{em}^2}{2^n}, \end{aligned}$$

where  $\mathcal{B}_1$  outputs a list of length  $2q_2q_{em}$ ,  $\mathcal{B}_2$  a list of length  $2q_{em}^2$ ,  $\mathcal{B}_3$  a list of length  $q_{em}^2$ , and  $\mathcal{B}_4$  a list of length at most  $q_{em}^2$ .  $\blacksquare$

## 4.6 Discussion

---

### 4.5.2.2 $\Phi^\oplus$ -RKA security

Corollary 4.4 together with Theorem 4.6 allow us to establish that the three-round single-key Even–Mansour construction with independent round permutations is  $\Phi^\oplus$ -RKCCA secure:

**Corollary 4.7.** *For an adversary  $\mathcal{A}$  (with parameters defined as before and  $\ell \leq 2^n/4$ ) against the  $\Phi^\oplus$ -RKCCA security of  $\text{EM}^\pi[1, 1, 1, 1]$ , we have that*

$$\text{Adv}_{\text{EM}^\pi[1,1,1,1],\Phi^\oplus,3}^{\text{rkcca}} \leq \frac{4q_{em}(q_2 + q_{em})(q_1 + q_3 + 3)}{2^n} + \frac{2q_{em}q_2}{2^n - q_2}.$$

Once again, via a direct analysis (but at the expense of modularity) the cubic bound above can be tightened to a quadratic one.

## 4.6 Discussion

In this chapter, we have given three strong, positive results about the security of Even–Mansour ciphers under related key attacks. It is interesting to consider the open questions which remain and possible future directions for research in this area.

An obvious open question is whether similar results can be obtained for other block cipher design strategies. Barbosa and Farshim have given a positive answer in the case of Feistel networks, but the answer is not known for Lai–Massey ciphers, Misty-like ciphers or other generalisations of Feistel ciphers.

A second question is whether the results in this chapter can be generalised to cover modifications to the Even–Mansour scheme. Dunkelman, Keller, and Shamir [98] consider several variants of the Even–Mansour scheme, including addition and involution Even–Mansour ciphers (where the xors are replaced with modular additions or the random permutations are replaced with random involutions, respectively). It appears straightforward to apply the techniques used in this chapter to obtain results about these schemes, although we have not worked through the precise details.

Another possible variant of the Even–Mansour scheme is one in which the same permutation is reused across the rounds. If the same permutation is used in the first

## 4.6 Discussion

---

and third rounds, a similar proof strategy applies as these oracles would be faithfully simulated in the reduction. The proof, however, does not immediately apply if the same permutation is used in all rounds as a forgetful simulation of the middle oracle introduces inconsistencies across the rounds.

Without going into the details, a proof can be obtained by introducing a new CF-type assumption which requires that it is infeasible to find  $(\phi_1, x_1, \phi_2, x_2)$  such that  $\phi_2(k) \oplus x_2 = P^\pm(x_1 \oplus \phi_1(k)) \oplus \phi_1(k)$ ; this condition would ensure that inconsistencies resulting from a  $P$  query in the first or third rounds and a  $S$  query happen with low probability. Following the proof of Proposition 4.3, we can also reduce this new property to standard OUP and CF notions: starting from the above winning condition, first consider the game where the winning condition uses an independent permutation (this change reduces to OUP), then consider the winning condition  $\phi_1(k) = \phi_2(k)$  (an adversary winning this game wins the CF game), finally if an adversary wins the second game but not the third, then they have found a solution to  $\phi_1(k) \oplus \phi_2(k) = x \oplus R$  where  $R$  is the random output of the independent permutation, which happens with probability at most  $\ell/(2^n - 2\ell)$  as  $x$ ,  $\phi_1(k)$ , and  $\phi_2(k)$  are independent of  $R$ . We remark that Biryukov and Wagner [49] describe slide attacks which can be applied to this construction, giving an upper bound on its security.

A further potential research direction to examine is the effect of more complex key schedules on our results. For example, it would be interesting to ask whether security under the wider range of RKD functions considered in this chapter can be established for schemes using the key schedules that Cogliati and Seurin [78] study.

A final open question is how to use these results when designing concrete proposals for block ciphers. We have shown that the three round Even–Mansour construction is sufficient for a wide range of uses, however the vast majority of schemes use many more rounds than this. The optimal trade off between the number of rounds and the complexity of the round functions, for given performance requirements, is not clear. An interesting question is therefore whether using ‘stronger’ round functions could significantly reduce the number of rounds required. AES<sup>2</sup> [60] uses this idea: it has only two rounds, but the round functions are full applications of AES using publicly known keys. We remark that Dinur et al. [96] have cryptanalysed this proposal (in the single-key model and not contradicting our bounds).

# The CLRW2 tweakable block cipher

---

## Contents

5.1	Introduction . . . . .	127
5.2	Preliminaries . . . . .	129
5.3	Proof summary . . . . .	131
5.4	The flaw in the proof . . . . .	139
5.5	A revised proof . . . . .	141
5.6	A limitation of this proof technique . . . . .	150

---

*Tweakable block ciphers are flexible primitives, which are being adopted as useful components from which to build efficient cryptographic schemes. In this chapter, we study Landecker et al.’s tweakable block cipher construction, CLRW2, and identify an error in the reduction given for this scheme. Fortunately, the issue can be resolved and a new bound for the security of CLRW2 is given. Additionally we identify a potential limitation of this proof technique when looking to extend the scheme to provide asymptotic security.*

*The work described in this chapter is available at [223].*

## 5.1 Introduction

Several recent proposals for symmetric encryption primitives, particularly AEAD schemes, feature tweakable block ciphers as part of their design. As discussed in Section 2.2.5, a tweakable block cipher is a block cipher that admits an additional public input (the ‘*tweak*’) to introduce extra variability at the message-block level.

## 5.1 Introduction

---

In their seminal paper on tweakable block ciphers, Liskov et al. [186, 187] describe the syntax and security requirements for tweakable block ciphers and describe several methods for building tweakable block ciphers from conventional block ciphers. There are a few dedicated designs for tweakable block ciphers [248, 87], however most research effort has focused on designing conventional block ciphers along with methods for transforming them into tweakable block ciphers (e.g. [238, 149]).

Many block-cipher-based encryption and authentication schemes are secure up to the birthday bound, i.e. provided that fewer than  $2^{\frac{n}{2}}$  queries are made, where  $n$  is the width of the block cipher (in bits). Beyond this point, one would expect the input to the block cipher to be repeated and for this to perhaps leak some information or simplify forgery attempts (as described in, for example, [23, 28, 30]). However for some applications, security beyond the birthday bound may be desirable, for example if large amounts of data are to be encrypted and the use of a block cipher with a larger block size is undesirable. Several works have studied the security of schemes beyond the birthday bound (e.g. [145, 151]); one related question is how to achieve beyond-birthday-bound security for tweakable block ciphers.

Minematsu [200] suggests a method to build a  $2n$ -bit tweakable block cipher that provides  $\mathcal{O}(2^{\frac{n+m}{2}})$  security from an  $n$ -bit block cipher, where  $m$  is the size of the tweak. This scheme has a Luby–Rackoff or Feistel structure, but has the disadvantage of only supporting short tweaks and requiring per-invocation rekeying of the block cipher which makes changing the tweak computationally expensive.

Landecker et al. [176] continue the study of tweakable block ciphers that remain secure beyond the birthday bound. They specify CLRW2, a tweakable block cipher construction (based on two copies of LRW2 [186]) which remains secure up to approximately  $2^{\frac{2n}{3}}$  queries; they also study the security of TBC-MAC (an analogue of CBC-MAC defined in terms of tweakable block ciphers). CLRW2 allows arbitrarily long tweaks and does not require excessive rekeying of the block cipher.

Lampe and Seurin [175] extend the CLRW2 construction and consider longer chains of the LRW2 construction and are able to show (asymptotically in the number of rounds, using a coupling argument) that this provides greater security further beyond the birthday bound than the CLRW2 construction. Their bounds agree

## 5.2 Preliminaries

---

with Landecker et al.'s bound for two rounds against non-adaptive chosen-plaintext attacks. They extend this result, additionally giving a weaker bound under chosen-ciphertext attacks. They conjecture that their non-adaptive chosen-plaintext bound also holds for chosen-ciphertext attacks.

Each of the constructions described above comes with a security proof. Recently, flaws have been found in the proofs given for two high-profile ciphers: GCM [148] and XCB [71]. In order for security proofs to provide meaningful guarantees and for the community to have faith in the proofs given for schemes, it is important that they are correct and any errors are removed. The factor lost by correcting the bound for CLRW2 is significantly smaller than for these schemes, however it remains important that any errors in a security reduction are removed.

### Contributions

In this chapter, we provide a detailed analysis of the proof given for CLRW2 [176]. We identify an error in the proof and are able to resolve it, giving more accurate bounds for the security offered by this construction. We also describe a potential limitation of this proof technique, which prevents it from being used to extend these results asymptotically.

Landecker et al. have independently identified and corrected the error in their proof [177, 252]; they correct the proof using a neat coupling argument which results in a tighter bound than is obtained in this chapter.

## 5.2 Preliminaries

This section contains a brief description of the CLRW2 scheme (further details are given in [176]) and some new notation used throughout this chapter.

## 5.2 Preliminaries

---

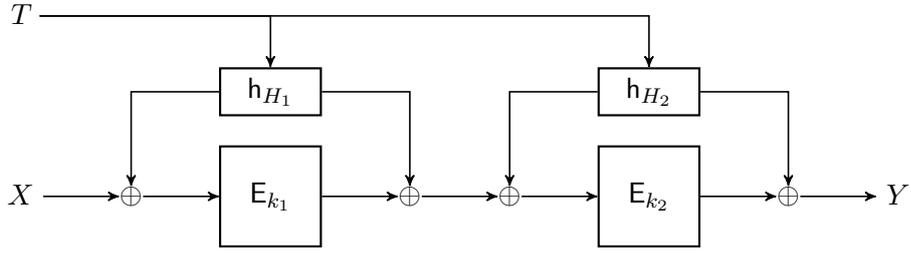


Figure 5.1: The CLRW2 tweakable block cipher construction.

### 5.2.1 Description of CLRW2

The scheme proposed by Landecker et al. [176] combines an  $\varepsilon$ -AXU hash function and a block cipher. The ciphertext  $Y$  is computed from plaintext  $X$  using key  $k = (k_1, H_1, k_2, H_2)$  and tweak  $T$  as follows:

$$Y = E_{k_2} (E_{k_1} (X \oplus h_{H_1}(T)) \oplus h_{H_1}(T) \oplus h_{H_2}(T)) \oplus h_{H_2}(T) .$$

This construction is illustrated in Figure 5.1. The intuition behind the security of CLRW2 is that an adversary can only obtain a birthday-bound style advantage by causing the input to both block ciphers to be repeated.

### 5.2.2 Notation

We largely follow the notation used by Landecker et al. [176] to avoid introducing confusion. Throughout we will use  $X$  and  $T$  to denote plaintext and tweak inputs to a tweakable block cipher respectively;  $Y$  will denote the output ciphertext. Queries made by an adversary and the value of random variables related to those queries are indexed by a counter  $i$ .

When using a random permutation (in place of a block cipher), we will lazily sample the random permutations instead of defining all pairs of input and output up front. When referring to the domain and range of a permutation  $\pi : \{0, 1\}^n \rightarrow \{0, 1\}^n$ , we will use  $\text{Dom}_{\text{full}}(\pi)$  and  $\text{Rng}_{\text{full}}(\pi)$  to denote the set  $\{0, 1\}^n$  in order to make clear the context that this set relates to. When lazy-sampling a permutation  $\pi : \{0, 1\}^n \rightarrow \{0, 1\}^n$  we will use the sets  $\text{Dom}_{\text{lazy}}(\pi) \subseteq \text{Dom}_{\text{full}}(\pi) = \{0, 1\}^n$  and  $\text{Rng}_{\text{lazy}}(\pi) \subseteq \text{Rng}_{\text{full}}(\pi) = \{0, 1\}^n$  to keep track of which values have been defined in

### 5.3 Proof summary

---

the domain and range (respectively) of  $\pi$ . We will often drop the subscript for the sets  $\text{Dom}_{\text{lazy}}(\pi)$  and  $\text{Rng}_{\text{lazy}}(\pi)$  provided that the meaning is clear. The ‘lazy’ sets have an implicit query index because they are only defined relative to the previous queries and random choices within them.

We reiterate that for CLRW2, a tweakable block cipher key includes two keys for the underlying block cipher and two keys for the universal hash function family, i.e.  $k = (k_1, H_1, k_2, H_2)$ . For simplicity and clarity, we abbreviate  $\text{h}_{H_j}$  to  $\text{h}_j$ .

### 5.3 Proof summary

In this section, we give a brief overview of Landecker et al.’s proof below but refer to the original paper [176] for the full details. The reduction given by Landecker et al. relies on the SPRP security of the block cipher and the universality of the hash function family to show that CLRW2 realises a strong tweakable block cipher (i.e. the TSPRP notion described in Section 2.2.5). The proof proceeds through a series of games in which the adversary interacts with a tweakable block cipher oracle, but not the individual component parts of the construction (the block ciphers and universal hash functions).

#### 5.3.1 Partitioning the output set

Landecker et al. partition  $\{0, 1\}^n$  (corresponding to points in the range of the tweakable block cipher) into four sets. The first step of the proof is to replace the block ciphers with permutations (chosen uniformly at random). The definition of the sets used to partition  $\{0, 1\}^n$  depends on the particular tweak value used in a query and the sets  $\text{Rng}_{\text{lazy}}(\pi_i)$ ; these sets therefore have an implicit query index.

Informally, we define  $\mathcal{Y}_i$  to be the set of possible output values from an ideal tweakable block cipher, given the responses to previous queries and recalling that each  $\text{TE}(k, T, \cdot)$  is a permutation. That is, we set  $\overline{\mathcal{Y}}_i = \{Y_j : j < i \text{ and } T_j = T_i\}$  and  $\mathcal{Y}_i = \{0, 1\}^n \setminus \overline{\mathcal{Y}}_i$ .

### 5.3 Proof summary

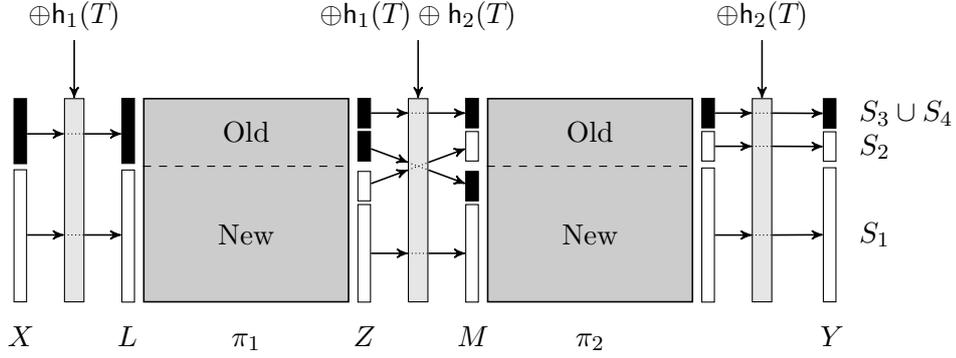


Figure 5.2: A graphical representation of the sets  $S_i$ . The domain and range of each permutation are divided into two sets: the input/output pairs that are ‘Old’ (in  $\text{Dom}(\pi_j)$  and  $\text{Rng}(\pi_j)$ ), shown above the dashed line; and the points that are ‘New’ (in  $\overline{\text{Dom}(\pi_j)}$  and  $\overline{\text{Rng}(\pi_j)}$ ), shown below the dashed line. For a query with  $L \notin \text{Dom}(\pi_1)$ , any value corresponding to a black box in the above diagram is impossible.

The set  $\{0, 1\}^n$  is then further partitioned as follows:

$$\begin{aligned}
 S_1 &= \{Y \in \mathcal{Y}_i : Y \oplus h_2(T) \notin \text{Rng}(\pi_2)\} , \\
 S_2 &= \{Y \in \mathcal{Y}_i : Y \oplus h_2(T) \in \text{Rng}(\pi_2) \text{ and} \\
 &\quad \pi_2^{-1}(Y \oplus h_2(T)) \oplus h_1(T) \oplus h_2(T) \notin \text{Rng}(\pi_1)\} , \\
 S_3 &= \{Y \in \mathcal{Y}_i : Y \oplus h_2(T) \in \text{Rng}(\pi_2) \text{ and} \\
 &\quad \pi_2^{-1}(Y \oplus h_2(T)) \oplus h_1(T) \oplus h_2(T) \in \text{Rng}(\pi_1)\} , \\
 S_4 &= \{Y \in \overline{\mathcal{Y}_i}\} .
 \end{aligned}$$

These sets are shown graphically in Figure 5.2. ‘Old’ values are those in  $\text{Dom}_{\text{lazy}}(\pi)$  and  $\text{Rng}_{\text{lazy}}(\pi)$ ; the behaviour of  $\pi$  on ‘New’ values has not yet been defined.

That is,  $S_1$  is the set of output values corresponding to undefined outputs from  $\pi_2$ . The elements of  $S_2$  correspond to defined outputs from  $\pi_2$  with undefined outputs from  $\pi_1$ . The set  $S_3$  contains output values for which the inputs and outputs to both block ciphers are defined. Finally,  $S_4 = \overline{\mathcal{Y}_i}$  is the set of values that are not possible for either an ideal tweakable block cipher or the CLRW2 construction; responding to a (non-repeated, non-redundant) query with an element of this set would violate the requirement that each  $\text{TE}(k, T, \cdot)$  is a permutation.

At a first glance,  $S_3$  appears to be the difference between CLRW2 and an ideal tweakable block cipher: elements in  $S_4$  are not possible in either case; elements in

### 5.3 Proof summary

---

$\begin{aligned} &\text{TE}(k, T, X): \\ &i \leftarrow i + 1; X_i \leftarrow X; T_i \leftarrow T \\ &L \leftarrow X \oplus \mathbf{h}_1(T) \\ &Z \leftarrow \mathbf{E}_{k_1}(L) \\ &M \leftarrow Z \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T) \\ &Y \leftarrow \mathbf{E}_{k_2}(M) \oplus \mathbf{h}_2(T) \\ &\text{Return } Y \end{aligned}$
--

Figure 5.3: The TE oracle in Game 0 for the proof of Theorem 5.3.

$S_1$  and  $S_2$  are possible in both cases; elements in  $S_3$  are not excluded from the output of an ideal tweakable block cipher but are impossible in the case that  $L$  is ‘New’ when CLRW2 is used.

This informal summary does not give the full detail—if the situation were this simple, the original proof would be correct. Additionally, it is necessary to ensure that the probability of each element appearing as the output from the tweakable block cipher (or its inverse) is approximately equal in both the real and ideal cases.

We use  $p_{TBC}(Y)$  to denote the probability that  $Y$  is the output of the ideal tweakable block cipher and  $p_{G_3}(Y)$  for the probability that  $Y$  is the output of the intermediate cipher defined in Game 3 of the proof; these probabilities are both conditioned on all previous queries and responses. Note that  $p_{TBC}(Y) = p_{TBC}(Y')$  and  $p_{G_3}(Y) = p_{G_3}(Y')$  if  $Y$  and  $Y'$  are in the same set  $S_i$ .

#### 5.3.2 Structure of the games

The proof given by Landecker et al. considers an adversary interacting with the TSPRP game defined in Section 2.2.5; a series of eight games is then defined in which the TE oracle is modified.

Figure 5.3 defines TE so that it realises the CLRW2 scheme. In Game 1, the block cipher is replaced with a random permutation, which is lazily sampled. The security of this scheme relies on the assumption that the block cipher used to instantiate the scheme is a strong pseudorandom permutation and an adversary’s advantage in distinguishing between Game 0 and Game 1 is bounded above by  $2\mathbf{Adv}_{\mathbf{E}}^{\text{SPRP}}(\mathcal{A})$ .

### 5.3 Proof summary

---

The behaviour of **TE** is further changed in the following games. There are two cases considered for each query, as  $\pi_1$  is lazily sampled: either  $\pi_1(L)$  is already defined or  $L$  is a fresh value.

Games 2 and 3 modify **TE** in the case that  $\pi_1(L)$  is already defined. In both games, the ciphertext is chosen uniformly at random (from the set of possible outputs  $\mathcal{Y}_i$ ) and the value of  $\pi_1(L)$  is defined in such a way that it remains consistent with this choice of ciphertext. If the definition of  $\pi_1(L)$  contradicts the permutivity of  $\pi_2$ , one of  $\text{bad}_1$  and  $\text{bad}_2$  is set to **true**. In Game 2,  $Y$  is redefined when a **bad** event occurs; this is not the case in Game 3.

The probability of these **bad** events occurring bounds the distinguishing advantage between Games 1 and 3. The analysis of these games ensures that queries which cause both (lazily sampled) permutations to be queried at a defined value are rare and do not significantly distort the distribution of output values from that of an ideal tweakable block cipher.

Games 4 through to 7 modify **TE** when  $\pi_1(L)$  has not been defined; no changes are made to the code that is executed when  $\pi_1(L)$  is already defined. It is in these games that we have identified the problem with the proof. The objective of these changes is to show that the ciphertext can be chosen uniformly from  $\mathcal{Y}_i$  without the probability distribution differing significantly from ideal. The main idea is to sample the ciphertext uniformly from one of the  $S_i$  for each query (using a biased coin to decide which  $S_i$ ) and then to show that the resulting ciphertext distribution is close to uniform on  $\mathcal{Y}_i$  (as would be the case for an ideal tweakable block cipher).

Again, **bad** flags are set to **true** if the permutivity of  $\pi_1$  or  $\pi_2$  is contradicted; the probability of these **bad** events occurring bounds the distinguishing advantage between Games 4 and 7. The analysis of these games leads to the same conclusion as the previous games, in the case that  $\pi_1(L)$  has not already been defined.

Finally, in Game 8, the adversary can choose the  $Y$  values that are returned for each query. The motivation for this step is that this makes it no harder for an adversary to trigger any of the **bad** events, but makes it easier to reason about the probability of **bad** events occurring.

### 5.3 Proof summary

---

Landecker et al. conclude that the scheme remains secure up to approximately  $2^{\frac{2n}{3}}$  queries, at which point one would expect to have observed a bad event.

#### 5.3.3 Games 4 and 5

The identified error is in the transition between Games 4 and 5, so it is useful to have a detailed description of these games. The games, as defined by Landecker et al., are given in Figures 5.4 and 5.5. Between these two games, the methods used to sample  $V$  and  $Y$  (and in particular, the order in which they are sampled) changes; the final joint distribution of  $V$  and  $Y$  should not change between these games, but we will see that this is not achieved.

We briefly describe the sampling techniques used to choose  $Y$  in the different games. In Game 3,  $Z$  is chosen uniformly at random from  $\overline{\text{Dom}(\pi_1)}$  (as one would expect, because  $\pi_1$  is lazily sampled) and  $Y$  is defined to be consistent with this choice of  $Z$ . In Game 4, an appropriately weighted coin is tossed and  $Y$  is chosen from either  $S_1$  or  $S_2$  depending on the outcome;  $Z$  is then defined consistently, so that the distributions of  $Z$  and  $Y$  are identical to the distribution in Game 3. In Game 5,  $Y$  is sampled from  $\mathcal{Y}_i$  before the output of tossing the weighted coin is known. This introduces the possibility that  $Y \in S_3$  (which is not possible in Game 4 if  $L \notin \text{Dom}(\pi_1)$ ); if  $Y$  is chosen in  $S_3$ , then **bad** is set and  $Y$  is resampled from either  $S_1$  or  $S_2$ . Game 6 is identical to Game 5, except that if  $Y \in S_3$  then it is not resampled.

Landecker et al. define  $\Delta_i = \sum_{Y \in S_i} |p_{G3}(Y) - p_{TBC}(Y)|$ , the absolute magnitude of the difference between the ideal probability that  $Y \in S_i$  and the actual probability realised when using CLRW2. They also define  $N = |\overline{\text{Dom}(\pi_1)}|$ . The distributions of ciphertext in Games 4, 5, and 6 are illustrated in Figures 5.6, 5.7, and 5.8.

### 5.3 Proof summary

---

Game 4:

TE( $T, X$ ):

$i \leftarrow i + 1; X_i \leftarrow X; T_i \leftarrow T$

$L \leftarrow X \oplus \mathbf{h}_1(T)$

If  $L \in \text{Dom}(\pi_1)$

$M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

$Y \leftarrow \mathcal{Y}_i$

If  $M \in \text{Dom}(\pi_2)$

$\text{bad}_1 \leftarrow \text{true}$

If  $Y \oplus \mathbf{h}_2(T) \in \text{Rng}(\pi_2)$

$\text{bad}_2 \leftarrow \text{true}$

$\pi_2(M) \leftarrow Y \oplus \mathbf{h}_2(T)$

Else

$V_i \leftarrow \xi(|S_2| / |\overline{\text{Rng}(\pi_1)}|)$

If  $V_i = 1$

$Y \leftarrow S_2$

If  $V_i = 0$

$Y \leftarrow S_1$

If  $Y \in S_2$

$Z \leftarrow \pi_2^{-1}(Y \oplus \mathbf{h}_2(T)) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

If  $Y \in S_1$

$Z \leftarrow \overline{\text{Rng}(\pi_1)} \setminus (\text{Dom}(\pi_2) \oplus \mathbf{h}_2(T) \oplus \mathbf{h}_1(T))$

$\pi_2(Z \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)) \leftarrow Y \oplus \mathbf{h}_2(T)$

$\pi_1(L) \leftarrow Z$

$M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

Return  $Y$

Figure 5.4: Game 4 as defined by Landecker et al.

### 5.3 Proof summary

---

**Game 5**, **Game 6:**

**TE**( $T, X$ ):

$i \leftarrow i + 1; X_i \leftarrow X; T_i \leftarrow T$

$L \leftarrow X \oplus \mathbf{h}_1(T)$

If  $L \in \text{Dom}(\pi_1)$

$M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

$Y \leftarrow_s \mathcal{Y}_i$

If  $M \in \text{Dom}(\pi_2)$

**bad**<sub>1</sub>  $\leftarrow$  true

If  $Y \oplus \mathbf{h}_2(T) \in \text{Rng}(\pi_2)$

**bad**<sub>2</sub>  $\leftarrow$  true

$\pi_2(M) \leftarrow Y \oplus \mathbf{h}_2(T)$

Else

$Y \leftarrow_s \mathcal{Y}_i$

If  $Y \in S_1$

$V_i \leftarrow 0$

If  $Y \in S_2$

$V_i \leftarrow 1$

If  $Y \in S_3$

**bad**<sub>3</sub>  $\leftarrow$  true

$V_i \leftarrow_s \xi(\Delta_2 / (\Delta_1 + \Delta_2))$

$Z \leftarrow \pi_2^{-1}(Y \oplus \mathbf{h}_2(T)) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

// In Game 6,  $V_i = \perp$ .

If  $V_i = 1$

$Y \leftarrow_s S_2$

If  $V_i = 0$

$Y \leftarrow_s S_1$

If  $Y \in S_2$

$Z \leftarrow \pi_2^{-1}(Y \oplus \mathbf{h}_2(T)) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

If  $Y \in S_1$

$Z \leftarrow_s \overline{\text{Rng}(\pi_1)} \setminus (\text{Dom}(\pi_2) \oplus \mathbf{h}_2(T) \oplus \mathbf{h}_1(T))$

$\pi_2(Z \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)) \leftarrow Y \oplus \mathbf{h}_2(T)$

$\pi_1(L) \leftarrow Z$

$M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)$

Return  $Y$

Figure 5.5: Games 5 and 6 as defined by Landecker et al. Boxed statements are included in Game 5 and are omitted from Game 6.

### 5.3 Proof summary

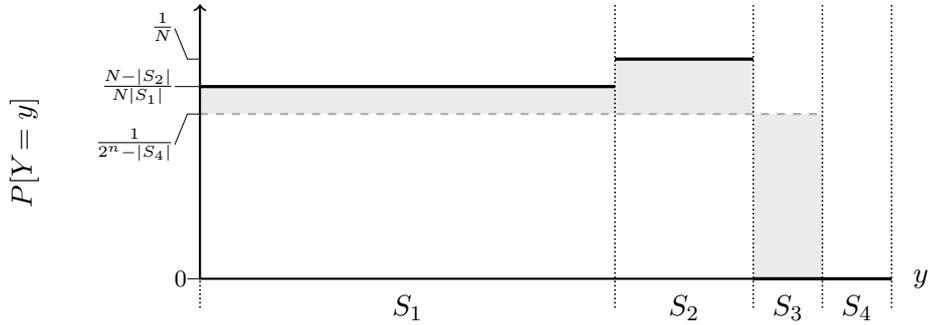


Figure 5.6: Ciphertext distribution in Game 4. The output of CLRW2 is denoted by solid, black lines. The dashed line indicates the behaviour of an ideal tweakable block cipher. The shaded areas correspond to  $\Delta_i$ .

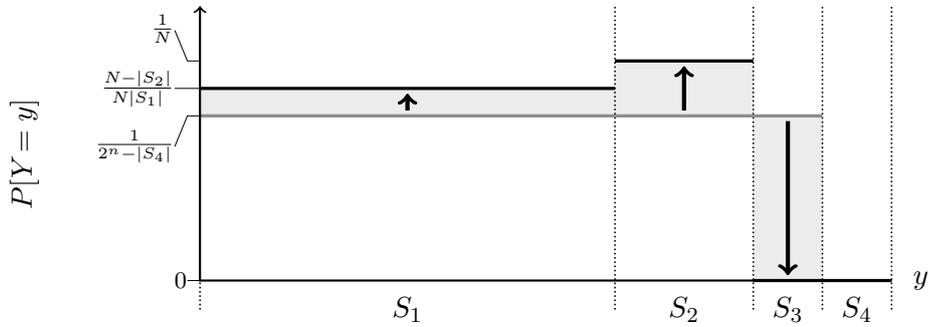


Figure 5.7: Ciphertext distribution in Game 5. The solid, grey line denotes the distribution from which the output is initially sampled. The output is resampled according to the bold arrows. The solid, black lines represent the final distribution, which is identical to that of Game 4.

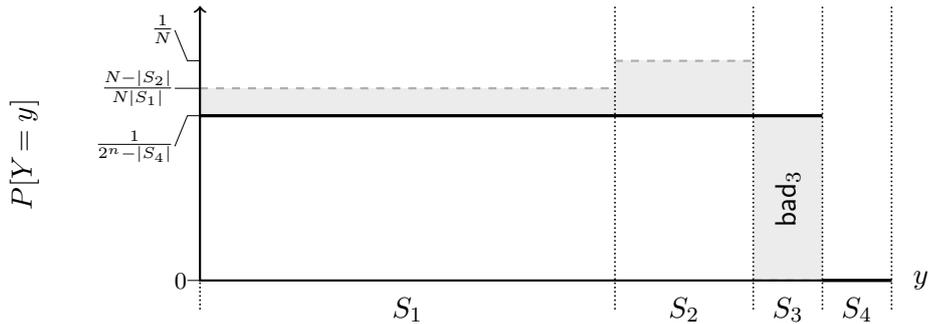


Figure 5.8: Ciphertext distribution in Game 6. The solid, black line denotes the distribution of outputs from CLRW2 in Game 6; this is the same as that for an ideal tweakable block cipher.  $\text{bad}_3$  is set if the output is sampled from the labelled region. The dashed line indicates the behaviour of CLRW2 in previous games; Game 6 is identical to Game 5 unless  $\text{bad}_3$  is set to true.

## 5.4 The flaw in the proof

---

### 5.4 The flaw in the proof

In the proof of security given for CLRW2 [176], Landecker et al. assert that the output distributions of Games 4 and 5 are identical. However, due to the way that  $Y$  is resampled if it is drawn from  $S_3$  in Game 5, this is only the case if  $p_{G3}(Y) - p_{TBC}(Y)$  is non-negative for  $Y \in S_1 \cup S_2$ . Unfortunately, it is possible for a series of queries to result in  $p_{G3}(Y) - p_{TBC}(Y)$  being negative for all  $Y \in S_1$ . This leads to a contradiction of the claim that the output distributions of Games 4 and 5 are always identical. We call the situation in which  $p_{G3}(Y) - p_{TBC}(Y) < 0$  for some  $Y \in S_i \cup S_2$  an inversion.

In order to describe the problem and fix the proof, we redefine  $\Delta_i$  as

$$\Delta_i = \sum_{Y \in S_i} p_{G3}(Y) - p_{TBC}(Y) ,$$

which lacks the modulus signs from Landecker et al.'s definition. It is worth noting that  $|\Delta_i| = \sum_{Y \in S_i} |p_{G3}(Y) - p_{TBC}(Y)|$  for each  $i$ , because for a given set  $S_j$   $p_{G3}(Y) - p_{TBC}(Y)$  has the same sign for every  $Y$  in  $S_j$ . Using our definition,  $|\Delta_j|$  corresponds precisely with Landecker et al.'s definition of  $\Delta_j$ .

#### 5.4.1 Flawed sampling methods

We briefly describe the differences between the sampling methods employed when  $L$  is new in the relevant games. When  $Y$  is resampled in Game 5, the probability that it is chosen in  $S_2$  is  $\frac{|\Delta_2|}{|\Delta_1| + |\Delta_2|}$ . This probability is used because if  $|\Delta_1| + |\Delta_2| = |\Delta_3|$  then the distribution of  $Y$  does not change between Games 4 and 5. However, if  $p_{G3}(Y) - p_{TBC}(Y) < 0$  then  $|\Delta_2| = |\Delta_1| + |\Delta_3|$ , so  $|\Delta_1| + |\Delta_2| > |\Delta_3|$  and the distribution of  $Y$  does change.

In fact, the difference between the distributions in Games 4 and 5 is exaggerated by the method used by Landecker et al. to resample from  $S_1 \cup S_2$  if  $Y \in S_3$ , as illustrated in Figure 5.13. In the case that  $p_{G3}(Y) - p_{TBC}(Y) < 0$  for  $Y \in S_1$ , the desired difference between the distributions in Game 4 and Game 5 for  $Y \in S_1$  is  $-|\Delta_1|$ , i.e.

$$\sum_{Y \in S_1} p_{G3}(Y) = \sum_{Y \in S_1} p_{TBC}(Y) - |\Delta_1| ,$$

## 5.4 The flaw in the proof

---

but using the sampling method described, this increases to  $|\Delta_1| \cdot \frac{|\Delta_3|}{|\Delta_1|+|\Delta_2|}$ , so that

$$\sum_{Y \in S_1} p_{G3}(Y) = \sum_{Y \in S_1} p_{TBC}(Y) + |\Delta_1| \cdot \frac{|\Delta_3|}{|\Delta_1| + |\Delta_2|} .$$

Similarly, the difference for  $Y \in S_2$  is decreased from  $|\Delta_2|$  to  $|\Delta_2| \cdot \frac{|\Delta_3|}{|\Delta_1|+|\Delta_2|}$ .

### 5.4.2 Causing an inversion

The proof given by Landecker et al. [176] is correct provided no inversions occur. We now describe how an adversary can, with high probability, cause an inversion.

First, recall that:

$$\begin{aligned} p_{TBC}(Y) &= \frac{1}{2^n - |S_4|} && \text{for } Y \notin S_4 , \\ p_{G3}(Y) &= \frac{N - |S_2|}{N|S_1|} && \text{for } Y \in S_1 , \\ p_{G3}(Y) &= \frac{1}{N} && \text{for } Y \in S_2 , \end{aligned}$$

where  $N = |\overline{\text{Dom}(\pi_1)}|$ . Also, note that for  $Y \in S_1$ , an inversion occurs if

$$\frac{1}{2^n - |S_4|} = p_{TBC}(Y) > p_{G3}(Y) = \frac{N - |S_2|}{N|S_1|} .$$

We now consider the possibility of an inversion occurring for  $Y \in S_1$ ; in this case an adversary can cause an inversion to occur with high probability.

**Lemma 5.1.** *Suppose that the adversary asks a number of queries (with no restrictions on how  $X$  and  $T$  are chosen) so that  $|\text{Dom}(\pi_1)| = a$  and  $|\text{Dom}(\pi_2)| = b$ . If the adversary uses a new tweak for the next query,  $L$  is new, and for every  $Z \in \text{Rng}(\pi_1)$  we have that  $Z \oplus h_1(T) \oplus h_2(T) \in \overline{\text{Dom}(\pi_2)}$  then an inversion occurs.*

*Proof.* In this case:

$$N = 2^n - a, \quad |S_1| = 2^n - b, \quad |S_2| = b, \quad \text{and } |S_3| = |S_4| = 0 ,$$

## 5.5 A revised proof

---

and the following statements are equivalent:

$$\begin{aligned}
& ab > 0 , \\
& 2^{2n} - (a + b)2^n + ab > 2^{2n} - (a + b)2^n , \\
& (2^n - a)(2^n - b) > 2^n(2^n - a - b) , \\
& \frac{1}{2^n} > \frac{2^n - a - b}{(2^n - a)(2^n - b)} , \\
& \frac{1}{2^n - |S_4|} > \frac{N - |S_2|}{N|S_1|} , \\
& p_{TBC}(Y) > p_{G3}(Y) ,
\end{aligned}$$

which is the condition for an inversion. ■

This situation can occur and indeed it is easy for an adversary to force this event to happen. If  $T_1 \neq T_2$  then an inversion occurs on the second query (where  $a = b = 1$ ) with probability  $1 - \varepsilon \approx \frac{2^n - 1}{2^n}$  (which is the probability that  $h_j(T_1) \neq h_j(T_2)$  when  $T_1 \neq T_2$ ) provided that  $L \in \overline{\text{Dom}(\pi_1)}$  for the second query.

It is also possible to show that inversions never occur for  $Y \in S_2$ .

**Lemma 5.2.** *For every  $Y \in S_2$  and every adversary  $\mathcal{A}$ ,  $p_{G3}(Y) - p_{TBC}(Y) \geq 0$ .*

*Proof.* In this case, we would require that

$$\frac{1}{2^n - |S_4|} = p_{TBC}(Y) > p_{G3}(Y) = \frac{1}{N} .$$

However, note that

$$2^n - |S_4| \geq |S_1| + |S_2| \geq N ,$$

and so

$$\frac{1}{2^n - |S_4|} \leq \frac{1}{|S_1| + |S_2|} \leq \frac{1}{N} .$$

Therefore, for  $Y \in S_2$ , there is no situation in which  $p_{TBC}(Y) \geq p_{G3}(Y)$ . ■

## 5.5 A revised proof

The proof given by Landecker et al. can be fixed by modifying only Games 5 to 8; in doing so, we do not deviate from their proof strategy. This leads to the

## 5.5 A revised proof

---

following theorem. We remark that Landecker et al. have independently identified an alternative method to correct the error in their proof [177, 252], using a coupling argument resulting in a tighter bound than is achieved here.

**Theorem 5.3.** *Let  $\mathcal{A}$  be an adversary asking a total of  $q$  queries to its oracles. Consider the CLRW2 construction instantiated with an  $\varepsilon$ -AXU hash function. Let  $\hat{\varepsilon} = \max\{1/(2^n - 2q), \varepsilon\}$ . Then there exists an adversary  $\mathcal{B}$  using the same resources, such that*

$$\mathbf{Adv}_{\text{TBC}}^{\text{tsprp}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{BC}}^{\text{sprp}}(\mathcal{B}) + \frac{8q^3 \varepsilon^2}{1 - q^3 \varepsilon^2}.$$

This should be contrasted with the original result, which concludes that there is an adversary  $\mathcal{B}$  against the SPRP security of  $\mathbf{E}$ , such that:

$$\mathbf{Adv}_{\text{TBC}}^{\text{tsprp}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{BC}}^{\text{sprp}}(\mathcal{B}) + \frac{6q^3 \varepsilon^2}{1 - q^3 \varepsilon^2}.$$

*Proof.* The idea of the proof is to leave Games 1 to 4 unchanged from Landecker et al.'s description, but to sample  $Y$  in Game 5 in such a way as to ensure that the distribution of ciphertexts in Games 4 and 5 remain identical, even in the presence of inversions (and propagate this modification forward through the later games).

This requires us to reduce the probability of  $Y$  being sampled from  $S_1$  when  $\Delta_1 < 0$  in Game 5; we do this naively by tossing an appropriately weighted coin to decide whether to resample  $Y$  from  $S_2$ . We add conditional branches to differentiate between the cases  $\Delta_1 \geq 0$  and  $\Delta_1 < 0$ ; this is a simple approach, but appears to work well and we lose only a small factor in the bound.

As before, the proof proceeds through 8 games; the revised versions of Games 5 to 8 are given in Figures 5.9, 5.10, and 5.11. We have only specified the encryption algorithm for each of game; it is straightforward to derive the corresponding decryption algorithms. The ciphertext distributions realised by each of these games if an inversion occurs are graphically represented in Figures 5.12, 5.13, and 5.14; in the absence of an inversion, the distributions are identical to those illustrated for Landecker et al.'s original proof in Figures 5.6, 5.7, and 5.8.

## 5.5 A revised proof

<p><b>Game 5'</b>, <b>Game 6'</b>:</p> <p><math>\text{TE}(T, X)</math>:</p> <p><math>i \leftarrow i + 1; X_i \leftarrow X; T_i \leftarrow T</math></p> <p><math>L \leftarrow X \oplus \mathbf{h}_1(T)</math></p> <p>If <math>L \in \text{Dom}(\pi_1)</math></p> <p style="padding-left: 20px;"><math>M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)</math></p> <p style="padding-left: 20px;"><math>Y \leftarrow_s \mathcal{Y}_i</math></p> <p style="padding-left: 20px;">If <math>M \in \text{Dom}(\pi_2)</math></p> <p style="padding-left: 40px;"><math>\text{bad}_1 \leftarrow \text{true}</math></p> <p style="padding-left: 20px;">If <math>Y \oplus \mathbf{h}_2(T) \in \text{Rng}(\pi_2)</math></p> <p style="padding-left: 40px;"><math>\text{bad}_2 \leftarrow \text{true}</math></p> <p style="padding-left: 40px;"><math>\pi_2(M) \leftarrow Y \oplus \mathbf{h}_2(T)</math></p> <p>Else <math>Y \leftarrow_s \mathcal{Y}_i</math></p>	
<p>If <math>\Delta_1 \geq 0</math></p> <p style="padding-left: 20px;">If <math>Y \in S_1</math></p> <p style="padding-left: 40px;"><math>V_i \leftarrow 0</math></p> <p style="padding-left: 20px;">If <math>Y \in S_2</math></p> <p style="padding-left: 40px;"><math>V_i \leftarrow 1</math></p> <p style="padding-left: 20px;">If <math>Y \in S_3</math></p> <p style="padding-left: 40px;"><math>\text{bad}_3 \leftarrow \text{true}</math></p> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 5px auto;"> <math>V_i \leftarrow_s \xi\left(\frac{ \Delta_2 }{ \Delta_1  +  \Delta_2 }\right)</math> </div> <p style="padding-left: 20px;"><math>Z \leftarrow \pi_2^{-1}(Y \oplus \mathbf{h}_2(T)) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)</math></p> <p style="padding-left: 20px;">If <math>V_i = 1</math></p> <p style="padding-left: 40px;"><math>Y \leftarrow_s S_2</math></p> <p style="padding-left: 20px;">If <math>V_i = 0</math></p> <p style="padding-left: 40px;"><math>Y \leftarrow_s S_1</math></p>	<p>If <math>\Delta_1 &lt; 0</math></p> <p style="padding-left: 20px;">If <math>Y \in S_3</math></p> <p style="padding-left: 40px;"><math>\text{bad}_3 \leftarrow \text{true}</math></p> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 5px auto;"> <math>Y \leftarrow_s S_2</math> </div> <p style="padding-left: 20px;">If <math>Y \in S_1</math></p> <p style="padding-left: 40px;"><math>V_i \leftarrow 0</math></p> <p style="padding-left: 40px;"><math>U_i \leftarrow_s \xi\left(\frac{ \Delta_1 (2^n -  S_4 )}{ S_1 }\right)</math></p> <p style="padding-left: 40px;">If <math>U_i = 1</math></p> <p style="padding-left: 80px;"><math>\text{bad}_4 \leftarrow \text{true}</math></p> <div style="border: 1px solid black; padding: 2px; width: fit-content; margin: 5px auto;"> <math>Y \leftarrow_s S_2</math> </div> <p style="padding-left: 20px;">If <math>Y \in S_2</math></p> <p style="padding-left: 40px;"><math>V_i \leftarrow 1</math></p>
<p>If <math>Y \in S_2</math></p> <p style="padding-left: 20px;"><math>Z \leftarrow \pi_2^{-1}(Y \oplus \mathbf{h}_2(T)) \oplus \mathbf{h}_2(T) \oplus \mathbf{h}_1(T)</math></p> <p>Else If <math>Y \in S_1</math></p> <p style="padding-left: 20px;"><math>Z \leftarrow_s \overline{\text{Rng}(\pi_1)} \setminus (\text{Dom}(\pi_2) \oplus \mathbf{h}_2(T) \oplus \mathbf{h}_1(T))</math></p> <p style="padding-left: 40px;"><math>\pi_2(Z \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)) \leftarrow Y \oplus \mathbf{h}_2(T)</math></p> <p style="padding-left: 20px;"><math>\pi_1(L) \leftarrow Z</math></p> <p style="padding-left: 20px;"><math>M \leftarrow \pi_1(L) \oplus \mathbf{h}_1(T) \oplus \mathbf{h}_2(T)</math></p> <p>Return <math>Y</math></p>	

Figure 5.9: Games 5' and 6' in the proof of Theorem 5.3. Boxed statements are included in Game 5' and are omitted from Game 6'. Note that in Game 6' when  $\Delta_1 \geq 0$  and  $Y \in S_3$ ,  $V_i$  is not defined so neither of the following conditional branches are executed. Between Game 4 and Game 5', the order in which  $V_i$  and  $Y$  are sampled is reversed. Game 5' is identical to Game 5 and Game 6' is identical to Game 6, except for the addition of the conditional branch for the case where  $\Delta_1 < 0$ .

## 5.5 A revised proof

---

```

Game 7':
TE(T, X):
i ← i + 1; Xi ← X; Ti ← T
Y ←s  $\mathcal{Y}_i$ 
L ← X ⊕ h1(T)
If L ∈ Dom( $\pi_1$ )
  M ←  $\pi_1$ (L) ⊕ h1(T) ⊕ h2(T)
  If M ∈ Dom( $\pi_2$ )
    bad1 ← true
  If Y ⊕ h2(T) ∈ Rng( $\pi_2$ )
    bad2 ← true
     $\pi_2$ (M) ← Y ⊕ h2(T)
Else
  If Y ∈ S1
    Vi ← 0
    If  $\Delta_1 < 0$ 
      Ui ←s  $\xi\left(\frac{|\Delta_1|(2^n - |S_4|)}{|S_1|}\right)$ 
      If Ui = 1
        bad4 ← true
  If Y ∈ S2
    Vi ← 1
  If Y ∈ S3
    bad3 ← true
    Z ←  $\pi_2^{-1}(Y \oplus h_2(T)) \oplus h_1(T) \oplus h_2(T)$ 
  If Vi = 1
    Z ←  $\pi_2^{-1}(Y \oplus h_2(T)) \oplus h_2(T) \oplus h_1(T)$ 
  If Vi = 0
    Z ←s  $\overline{\text{Rng}(\pi_1)} \setminus (\text{Dom}(\pi_2) \oplus h_2(T) \oplus h_1(T))$ 
     $\pi_2(Z \oplus h_1(T) \oplus h_2(T)) \leftarrow Y \oplus h_2(T)$ 
   $\pi_1(L) \leftarrow Z$ 
  M ←  $\pi_1(L) \oplus h_1(T) \oplus h_2(T)$ 
Return Y

```

Figure 5.10: Game 7' in the proof of Theorem 5.3. The distributions of random variables in Games 6' and 7' are identical, with Game 7' simplifying some of the program flow. Game 7' is identical to Game 7, except for the addition of the conditional branch for the case where  $\Delta_1 < 0$ .

## 5.5 A revised proof

---

Game 8'

TE( $T, X, Y$ ):

$i \leftarrow i + 1; X_i \leftarrow X; T_i \leftarrow T$

$Y_i \leftarrow Y$

$L \leftarrow X \oplus h_1(T)$

If  $L \in \text{Dom}(\pi_1)$

$M \leftarrow \pi_1(L) \oplus h_1(T) \oplus h_2(T)$

If  $M \in \text{Dom}(\pi_2)$

$\text{bad}_1 \leftarrow \text{true}$

If  $Y \oplus h_2(T) \in \text{Rng}(\pi_2)$

$\text{bad}_2 \leftarrow \text{true}$

$\pi_2(M) \leftarrow Y \oplus h_2(T)$

Else

If  $Y \in S_1$

If  $\Delta_1 < 0$

$U_i \leftarrow_s \xi\left(\frac{|\Delta_1|(2^n - |S_4|)}{|S_1|}\right)$

If  $U_i = 1$

$\text{bad}_4 \leftarrow \text{true}$

$Z \leftarrow_s \overline{\text{Rng}(\pi_1)} \setminus (\text{Dom}(\pi_2) \oplus h_2(T) \oplus h_1(T))$

$\pi_2(Z \oplus h_1(T) \oplus h_2(T)) \leftarrow Y \oplus h_2(T)$

If  $Y \in S_2$

$Z \leftarrow \pi_2^{-1}(Y \oplus h_2(T)) \oplus h_1(T) \oplus h_2(T)$

If  $Y \in S_3$

$\text{bad}_3 \leftarrow \text{true}$

$Z \leftarrow \pi_2^{-1}(Y \oplus h_2(T)) \oplus h_1(T) \oplus h_2(T)$

$\pi_1(L) \leftarrow Z$

$M \leftarrow \pi_1(L) \oplus h_1(T) \oplus h_2(T)$

Return  $Y$

Figure 5.11: Game 8' in the proof of Theorem 5.3. This game gives the adversary control over  $Y$  values, so the 'bad flags' can be set at least as easily as they can in Game 7'. Game 8' is identical to Game 8, except for the addition of the conditional branch for the case where  $\Delta_1 < 0$ .

## 5.5 A revised proof

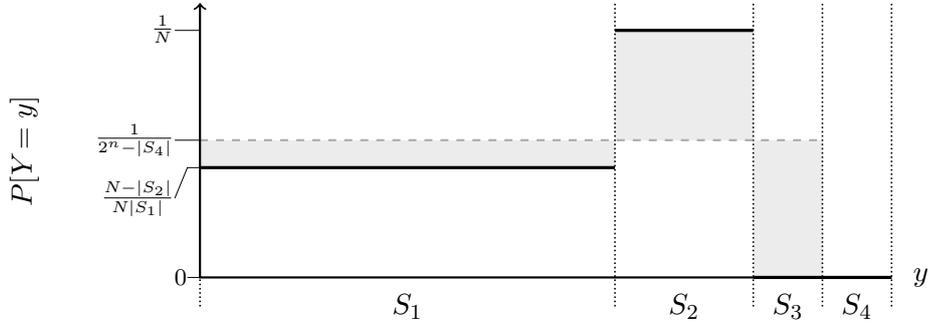


Figure 5.12: Ciphertext distribution in Game 4 when  $\Delta_1 \leq 0$ . The output of CLRW2 is denoted by solid, black lines. The dashed line indicates the behaviour of an ideal tweakable block cipher. We do not need to redefine Game 4 when  $\Delta_1 \geq 0$ .

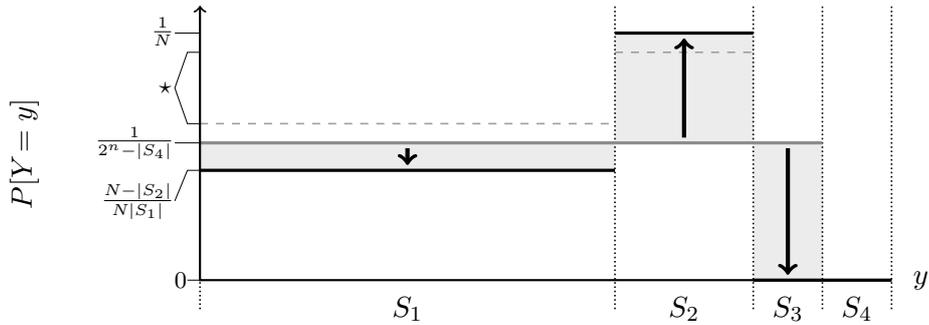


Figure 5.13: Ciphertext distribution in Game 5' when  $\Delta_1 < 0$ . The solid, grey line denotes the distribution from which the output is initially sampled; this is the output distribution according to an ideal tweakable block cipher. The output is resampled according to the bold arrows. The solid, black lines represents the final distribution, which is identical to that of Game 4. The dashed lines labelled by  $\star$  indicate the incorrect probabilities realised in Game 5 from Landecker et al.'s paper if  $\Delta_1 < 0$ .

## 5.5 A revised proof

---

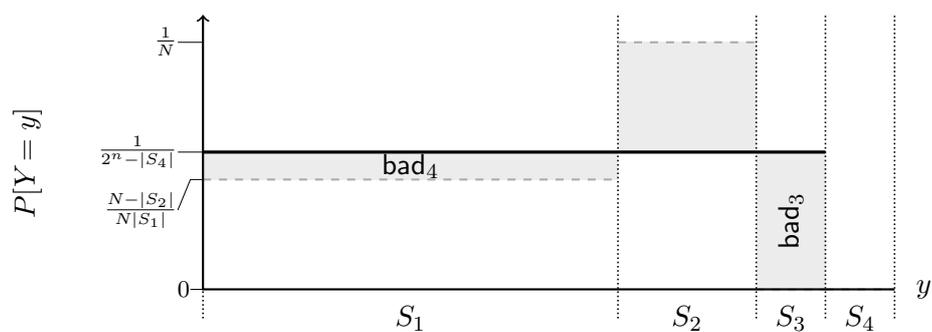


Figure 5.14: Ciphertext distribution in Game 6', when  $\Delta_1 < 0$ . The solid, black line denotes the distribution of CLRW2 in Game 6, which coincides with the distribution of outputs from an ideal tweakable block cipher.  $\text{bad}_3$  and  $\text{bad}_4$  are set to true if the output is sampled from the respectively labelled regions.

## 5.5 A revised proof

---

It remains to bound the maximum distinguishing advantage available to an adversary between each of these games. As Games 1 to 4 have not been modified, Landecker et al.'s original analysis still applies for this part of the proof. Thus,

$$\Pr [\mathcal{A}^{G^1} \rightarrow 1] \leq \Pr [\mathcal{A}^{G^4} \rightarrow 1] + \Pr [\mathcal{A}^{G^4} : \text{bad}_1 \vee \text{bad}_2] .$$

The distributions of all random variables in Games 4 and 5' are identical, so

$$\Pr [\mathcal{A}^{G^4} \rightarrow 1] + \Pr [\mathcal{A}^{G^4} : \text{bad}_1 \vee \text{bad}_2] \leq \Pr [\mathcal{A}^{G^{5'}} \rightarrow 1] + \Pr [\mathcal{A}^{G^{5'}} : \text{bad}_1 \vee \text{bad}_2] .$$

Games 5' and 6' are identical unless either  $\text{bad}_3$  or  $\text{bad}_4$  gets set to true, meaning that

$$\begin{aligned} \Pr [\mathcal{A}^{G^{5'}} \rightarrow 1] &\leq \Pr [\mathcal{A}^{G^{6'}} \rightarrow 1] + \Pr [\mathcal{A}^{G^{6'}} : \text{bad}_3 \vee \text{bad}_4] , \\ \Pr [\mathcal{A}^{G^{5'}} : \text{bad}_1 \vee \text{bad}_2] &\leq \Pr [\mathcal{A}^{G^{6'}} : (\text{bad}_1 \vee \text{bad}_2)] + \Pr [\mathcal{A}^{G^{6'}} : (\text{bad}_3 \vee \text{bad}_4)] . \end{aligned}$$

The distributions of random variables in Games 6' and 7' are identical, with Game 7' simplifying some of the program flow. Finally, Game 8' gives the adversary control over  $Y$  values, so the  $\text{bad}$  flags can be set at least as easily as they can in Game 7'. Thus

$$\begin{aligned} \Pr [\mathcal{A}^{G^1} \rightarrow 1] &\leq \Pr [\mathcal{A}^{G^{8'}} \rightarrow 1] + \Pr [\mathcal{A}^{G^{8'}} : (\text{bad}_1 \vee \text{bad}_2)] \\ &\quad + 2 \Pr [\mathcal{A}^{G^{8'}} : \text{bad}_3 \vee \text{bad}_4] . \end{aligned}$$

Noticing that  $\text{bad}_1$ ,  $\text{bad}_2$ , and  $\text{bad}_3$  are set in identical conditions in our revised games and Landecker et al.'s original games, we reuse their analysis:

$$\Pr [\mathcal{A}^{G^{8'}} : \text{bad}_1 \vee \text{bad}_2] \leq \frac{2q^3\hat{\varepsilon}^2}{1 - q^3\hat{\varepsilon}^2} \quad \text{and} \quad \Pr [\mathcal{A}^{G^{8'}} : \text{bad}_3] \leq \frac{2q^3\hat{\varepsilon}^2}{1 - q^3\hat{\varepsilon}^2} .$$

In order to bound the probability of an adversary setting  $\text{bad}_4$ , we need to bound  $|\Delta_1|^{\frac{(2^n - |S_4|)}{|S_1|}}$  in the case that  $\Delta_1 < 0$ . Lemma 5.4 gives us that

$$|\Delta_1|^{\frac{(2^n - |S_4|)}{|S_1|}} \leq \frac{q^3\hat{\varepsilon}^2}{1 - q^3\hat{\varepsilon}^2} .$$

As the adversary in Game 8' is interacting with an ideal tweakable block cipher, we can conclude that:

$$\Pr [\mathcal{A}^{G^1} \rightarrow 1] \leq \Pr [\mathcal{A}^{\text{iTBC}} \rightarrow 1] + \frac{8q^3\hat{\varepsilon}^2}{1 - q^3\hat{\varepsilon}^2} .$$

## 5.5 A revised proof

---

Thus, there is an adversary  $\mathcal{B}$  against the SPRP security of  $\mathbf{E}$ , such that:

$$\mathbf{Adv}_{\text{TBC}}^{\text{tsprp}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{BC}}^{\text{sprp}}(\mathcal{B}) + \frac{8q^3\varepsilon^2}{1 - q^3\varepsilon^2},$$

as required. ■

As mentioned above, bounding  $|\Delta_1| \frac{(2^n - |S_4|)}{|S_1|}$  in the case that  $\Delta_1 < 0$  bounds the advantage an adversary gains when we change from Game 5' to Game 6'.

**Lemma 5.4.** *Using the notation of Theorem 5.3, if  $\Delta_1 < 0$  then we have that*

$$|\Delta_1| \cdot \frac{(2^n - |S_4|)}{|S_1|} \leq \frac{q^2}{(2^n - q)^2} \leq \frac{q^3\varepsilon^2}{1 - q^3\varepsilon^2}.$$

*Proof.* We can bound  $|\Delta_1|$  in the case that  $\Delta_1 < 0$  as follows. Begin by noting that:

$$N \geq 2^n - q, \quad |S_1| \geq 2^n - q, \quad |S_2| \leq q, \quad |S_3| \geq 0, \quad \text{and} \quad |S_4| \geq 0.$$

It is straightforward to see that

$$\begin{aligned} |\Delta_1| &= \frac{|S_1|}{2^n - |S_4|} - \frac{N - |S_2|}{N} \\ &= \frac{|S_1|}{2^n - |S_4|} - 1 + \frac{|S_2|}{N} \\ &= \frac{|S_1| - (2^n - |S_4|)}{2^n - |S_4|} + \frac{|S_2|}{N} \\ &= \frac{-|S_2| - |S_3|}{2^n - |S_4|} + \frac{|S_2|}{N}. \end{aligned}$$

Applying the bounds noted above, we can derive that

$$\begin{aligned} \frac{-|S_2| - |S_3|}{2^n - |S_4|} + \frac{|S_2|}{N} &\leq -\frac{|S_2| + |S_3|}{2^n} + \frac{|S_2|}{N} \\ &= |S_2| \left( \frac{1}{N} - \frac{1}{2^n} \right) - \frac{|S_3|}{2^n} \\ &\leq |S_2| \left( \frac{1}{N} - \frac{1}{2^n} \right) \\ &\leq q \left( \frac{1}{N} - \frac{1}{2^n} \right) \\ &\leq q \left( \frac{1}{2^n - q} - \frac{1}{2^n} \right) \\ &= \frac{q^2}{2^n(2^n - q)}, \end{aligned}$$

## 5.6 A limitation of this proof technique

---

and hence that

$$\begin{aligned} |\Delta_1| \cdot \frac{(2^n - |S_4|)}{|S_1|} &\leq \frac{q^2}{2^n(2^n - q)} \cdot \frac{(2^n - |S_4|)}{|S_1|} \\ &\leq \frac{q^2}{2^n(2^n - q)} \cdot \frac{2^n}{(2^n - q)} \\ &= \frac{q^2}{(2^n - q)^2} . \end{aligned}$$

Finally, noting that  $\frac{1}{2^n - q} \leq \frac{1}{2^n - 2q} \leq \hat{\varepsilon}$  and that  $1 - q^3 \hat{\varepsilon}^2 \leq 1$ , it can be seen that

$$\frac{q^2}{(2^n - q)^2} \leq q^2 \hat{\varepsilon}^2 \leq \frac{q^3 \hat{\varepsilon}^2}{1 - q^3 \hat{\varepsilon}^2} ,$$

as required. ■

The first inequality in the statement of Lemma 5.4 is tight, in the sense that it is possible for an adversary to ask a series of  $q$  queries and for

$$|\Delta_1| \cdot \frac{(2^n - |S_4|)}{|S_1|} = \frac{q^2}{(2^n - q)^2} .$$

For this to occur we require that  $|S_4| = |S_3| = 0$ ,  $|S_2| = q$ , and  $N = 2^n - q$ .

## 5.6 A limitation of this proof technique

A natural extension of the work of Landecker et al. [176] is to consider longer chains of the LRW2 construction, as per Lampe and Seurin's work [175]. The naïve approach to proving results in this case (which we emphasise is not the approach taken by Lampe and Seurin) would be to mimic Landecker et al.'s proof, increasing the number of sets  $S_i$  in order to describe where the last 'fresh' input to a permutation occurs. This approach fundamentally depends on the ability to sample from a set  $\overline{\text{Rng}}_{\text{lazy}}(\pi_i) \cap (\overline{\text{Dom}}_{\text{lazy}}(\pi_{i+1}) \oplus \mathbf{h}_i(T) \oplus \mathbf{h}_{i+1}(T))$ . If this set is ever empty, an adversary may make a query that cannot be answered using this method and a different proof technique is required.

We bound, by  $q < 2^{n-1}$ , the number of queries that may be asked before the sampling method described above fails. We emphasise that this does not constitute an error in Landecker et al.'s proof and is not an issue in Lampe and Seurin's work [175]; it simply prevents the naïve, asymptotic extension of Landecker et al.'s result.

## 5.6 A limitation of this proof technique

---

This bound is obtained as follows. First, note that for every  $j$ ,  $|\text{Dom}_{\text{lazy}}(\pi_j)| = |\text{Rng}_{\text{lazy}}(\pi_j)|$  after every query. Also for every tweak,  $\mathbf{h}_j(T) \oplus \mathbf{h}_{j+1}(T)$  defines a perfect matching  $\{0, 1\}^n \rightarrow \{0, 1\}^n$  (representing  $\text{Rng}_{\text{full}}(\pi_j)$  and  $\text{Dom}_{\text{full}}(\pi_{j+1})$ ). Consider, in parallel, all matchings (corresponding to all possible values of  $\mathbf{h}_j(T) \oplus \mathbf{h}_{j+1}(T)$ ). In responding to a query, we remove (again from every matching) up to two edges: one edge that matches the output of  $\pi_j$  and one edge matching the input to  $\pi_{j+1}$ .

For it to remain possible to respond to any later query, we must be able to sample from the each of the sets  $\overline{\text{Rng}_{\text{lazy}}(\pi_j)} \cap (\overline{\text{Dom}_{\text{lazy}}(\pi_{j+1})} \oplus \mathbf{h}_j(T) \oplus \mathbf{h}_{j+1}(T))$ . This means that every matching must have at least one edge remaining. If there is a matching with no remaining edges, then there is a value  $a$  for which one of the  $\overline{\text{Rng}_{\text{lazy}}(\pi_j)} \cap (\overline{\text{Dom}_{\text{lazy}}(\pi_{j+1})} \oplus a)$  is empty and thus, with this simulation strategy, we cannot respond to every query made using a tweak for which  $a = \mathbf{h}_j(T) \oplus \mathbf{h}_{j+1}(T)$ .

We can guarantee that  $\overline{\text{Rng}_{\text{lazy}}(\pi_j)} \cap (\overline{\text{Dom}_{\text{lazy}}(\pi_{j+1})} \oplus \mathbf{h}_j(T) \oplus \mathbf{h}_{j+1}(T))$  is not empty for every  $i$  and  $j$ , provided that  $q < 2^{n-1}$ . While it may remain possible to sample from this set beyond this bound, it is not guaranteed and depends on both the adversary's queries and the random choices in the lazy sampling of the functions.

This does not cause a problem until  $n-1$  block ciphers are chained together because up to that point security is only provided for  $q < 2^{n-1}$ . However if  $n-1$  block ciphers are chained together using the CLRW2 construction, then this issue can occur and it may not be possible to respond to queries using this method.

# Standardisation and cryptography

---

## Contents

6.1	Introduction . . . . .	152
6.2	The composition of ChaCha20 and Poly1305 . . . . .	154
6.3	Algebraic cryptanalysis and ISO/IEC 29167-15 . . . . .	164
6.4	Discussion . . . . .	185

---

*In this chapter, we study the security of two schemes that have been proposed for standardisation. The first is a composition of Bernstein’s ChaCha20 and Poly1305 proposed as an authenticated encryption scheme for use in IETF protocols; the second is an ultra-lightweight RFID authentication protocol proposed as part of ISO/IEC 29167. We conclude that the first is a secure authenticated encryption scheme, while the second can be catastrophically broken by algebraic attacks.*

*The work described in this chapter on ChaCha20 and Poly1305 is available at [224]. The work described in this chapter on ISO/IEC 29167 is joint work with Carlos Cid, Loïc Ferreira, and Matthew Robshaw; it appears as [74]. All experiments were performed by the author.*

## 6.1 Introduction

Cryptographic schemes typically enable a number of parties to achieve some security objective: the classic example is two parties wanting to ensure confidentiality of their messages. Although this example only includes two parties, the ecosystem within which it exists consists of many users, possibly each with their own opinion on how

## 6.1 Introduction

---

to appropriately protect their data. Before they can communicate, they must first agree the precise details of the schemes that they will use. This includes a number of factors such as each party's intended security objectives, a choice of schemes claiming to achieve those objectives, and then parameters for those schemes.

In this situation, without some prior arrangement, it would be very unlikely that two parties would happen to have both implemented a protocol that they could use: in the extreme, they could find themselves agreeing on their security objectives and the algorithms that they wish to use, but a different choice of parameters (such as the choice of base-point on an elliptic curve) could leave them unable to communicate. Implementing cryptosystems comes at a non-zero cost, so there is merit in attempting to reduce the number of algorithms that must be implemented, without significantly reducing the likelihood that any two members of a community will be able to communicate. This is the objective of standardisation: defining a (small) number of schemes in order that two parties using those schemes gain some confidence that they will be able to interoperate successfully.

Because standardised schemes are likely to see widespread adoption, a critical feature of the standardisation process is that it must engender confidence in all parties. NIST's AES process is often given as an excellent example of how to standardise cryptosystems: proposals from a large number of teams were publicly and comprehensively evaluated for several years, before an open and transparent decision was reached. The SHA-3 process followed a similar formula. Unfortunately, this method is very intensive, requiring many parties to propose schemes and study the proposals.

There is plenty of demand for standardisation, due to the numerous potential applications for cryptography. This leads to a few techniques seeing widespread adoption, covering the majority of use cases. As an example, the original objective of the AES process was to standardise a block cipher for protecting sensitive government information in the United States [212], but it has been widely reused in a range of situations, in part because the process led to such trust in the algorithm.

For some applications, pre-existing standardised techniques are not appropriate: heavily resource-constrained devices may not have the memory, processors, power, or reliable randomness sources necessary to utilise these techniques; some environ-

## 6.2 The composition of ChaCha20 and Poly1305

---

ments may require a large degree of parallelism, high throughput or low latency; or there may be a desire to avoid dependence on a single scheme in case significant cryptanalytic progress requires a rapid transition to a stand-by scheme. In these situations, one could look to the output from projects such as eSTREAM to inform the decision [214]: although the objective was not to standardise schemes, it enjoyed involvement from a large group of academics, greatly expanded the understanding of stream ciphers, and concluded with a portfolio of novel, but well-studied designs. A final option, for when no existing standards or well-studied schemes meet the requirements, is to specify a new scheme; as we will see, this approach is not without its risks.

### Contributions

In this chapter we discuss two cryptosystems, both proposed for standardisation. The first is Langley’s novel combination [211] of Bernstein’s ChaCha20 stream cipher [35] and Poly1305 universal-hash-based message authentication code [34]. This combination was proposed for use in IETF protocols and has since been deployed as part of TLS. ChaCha20 and Poly1305 are both established algorithms and we give a security reduction demonstrating that this composition results in a secure authenticated encryption scheme, requiring standard assumptions on the primitives. The second scheme we consider is an RFID authentication protocol proposed as part of ISO/IEC 29167 [140]. This is a novel construction targeting ultra-lightweight devices; we demonstrate that it is insecure and can be catastrophically broken by algebraic attacks.

## 6.2 The composition of ChaCha20 and Poly1305

There has recently been a proposal to the Crypto Forum Research Group (CFRG) of the Internet Research Task Force (IRTF) to consider a combination of ChaCha20 and Poly1305 for inclusion in future internet protocols [211]. This proposal has come about, in part, due to concern over the reliance of existing IETF protocols on AES and the risk that advances in the cryptanalysis of AES could leave users without a good choice for a symmetric cryptographic primitive. A similar concern led to the

## 6.2 The composition of ChaCha20 and Poly1305

---

SHA-3 competition: improvements in attacks against SHA-1 caused NIST to transition to the SHA-2 family of hash functions and begin considering the specification of SHA-3. The SHA-3 competition aimed to identify an alternative hash function that would *‘improve the robustness of NIST’s hash algorithm toolkit’* [213].

Additionally, although GCM enjoys enviable performance on platforms that offer hardware acceleration for some of the component operations, it is less well suited to more constrained devices [178]. As these devices become more widespread, it is desirable to standardise a scheme that is better suited to this environment.

In this section, we give a reduction from the security of the proposed combination of ChaCha20 and Poly1305 to the PRF security of the ChaCha20 block function. Although both primitives are believed to be secure, it is possible to combine secure primitives in a way that is insecure (as described in Section 2.2.8 and [29]); for this reason, a security analysis of the combined scheme is important. The generic results of Bellare and Namprempre [29] do not apply in this case; they rely on independent keys being used in the component primitives.

The analysis presented in this section does not concern the security of the underlying primitives and we will assume that the ChaCha20 block function behaves as a PRF. The proof will also make use of the fact that Poly1305 is an  $\varepsilon$ -almost  $\Delta$  universal hash function [262] where  $\Delta$  represents addition modulo  $2^{128}$ .

### 6.2.1 Notation

For brevity, the ChaCha20 block function will be denoted by  $CC$ , Poly1305 by  $Poly$ , and the composition as defined in draft-irtf-cfrg-chacha20-poly1305-00 [211] by  $CC\&Poly$ . The key will be denoted by  $k$  and the initialisation vector (which must be a per-message nonce) will be denoted by  $N$ . Associated data, plaintext, and ciphertext will be denoted by  $A$ ,  $P$ , and  $C$  respectively.

The ChaCha20 block function outputs blocks of keystream that are 512 bits wide, so the plaintext and ciphertext will be encrypted and decrypted in blocks of 512 bits. These blocks of plaintext and ciphertext will be denoted  $P_i$  and  $C_i$  with

## 6.2 The composition of ChaCha20 and Poly1305

---

$P = P_1 || \dots || P_p$  and  $|P_i| = 512$  bits, except perhaps for  $P_p$  which may be shorter (similarly for  $C$ ).

Poly1305 operates on 128-bit blocks, so each block of 512 bits is further divided into 128-bit blocks, however the results in this section do not require us to consider this so we only parse strings into 512-bit blocks. When the strings  $A$  and  $C$  are zero padded to fill 128-bit blocks this will be denoted  $\bar{A}$  and  $\bar{C}$ ; padding is not required in ChaCha20, so there can be no confusion about the block size (either for parsing or padding). In the description of Poly1305,  $\text{len}_{A,C}$  will represent the 128-bit block consisting of a 64-bit representation of the length of  $A$  in bytes, concatenated with a 64-bit block corresponding to the length of  $C$ .

We use  $L_A$  and  $L_C$  to denote the maximum bit-lengths of associated data and ciphertexts that may be sent using CC&Poly according to the specification. Addition modulo  $2^{128}$  will be denoted by  $+$ , with the corresponding subtraction operation denoted by  $-$ .

### 6.2.2 Description of the algorithms

We now briefly describe the algorithms studied in this section. The internal details of these algorithms are not relevant to this note and readers are referred to papers such as [34] and [35] for further details.

#### 6.2.2.1 ChaCha20

ChaCha20 is a stream cipher proposed by Bernstein [35], which is designed following similar principles as Salsa20 [36] (an eSTREAM finalist). ChaCha20 generates a keystream by applying the ChaCha20 block function to the key, nonce, and a block counter, in a mode reminiscent of the counter mode of operation for block ciphers. Plaintext is then encrypted using this keystream, with block  $i$  of the plaintext xored with the output of the ChaCha20 block function, evaluated using block counter  $i$ .

## 6.2 The composition of ChaCha20 and Poly1305

---

The syntax of the ChaCha20 block function is

$$\text{CC} : \{0, 1\}^{256} \times \{0, 1\}^{32} \times \{0, 1\}^{96} \rightarrow \{0, 1\}^{512} .$$

That is, the ChaCha20 block function takes as input a 256-bit key, a 32-bit block number, and a 96-bit nonce, and outputs 512 pseudorandom bits.

ChaCha20 is widely believed to be secure, for example it is recommended by the European Union Agency for Network and Information Security (ENISA) [115]. A few papers have attacked reduced round versions for ChaCha20 (e.g. [13, 144, 250]), however this analysis has not contradicted its PRF security.

### 6.2.2.2 Poly1305

Poly1305 is a polynomial-based universal hash function (as studied in Chapter 3), also designed by Bernstein [34]. Some details were given in Sections 2.2.7 and 3.2.1.4; we re-iterate the relevant facts here. The syntax of Poly1305 is

$$\text{Poly} : \{0, 1\}^{128} \times \{0, 1\}^* \rightarrow \{0, 1\}^{128} ,$$

that is, Poly1305 takes as input a 128-bit key (which has some specific bits set to zero) and a message of arbitrary length, outputting a 128-bit string as the message digest. The output of Poly1305 is computed by evaluating a particular polynomial in  $\mathbb{F}_{2^{130-5}}$  (as described in Section 3.2.1) and then truncating the output to 128 bits. The key used in Poly1305 has certain bits ‘clamped’ to zero for performance reasons. The polynomial’s coefficients are determined by the message, with each 128-bit message block encoded to an integer modulo  $2^{130} - 5$ .

Bernstein [34] shows that Poly1305 is  $\varepsilon$ -almost  $\Delta$  universal where  $\Delta$  represents addition modulo  $2^{128}$ ,  $\varepsilon = \frac{8\lceil L/16 \rceil}{2^{106}}$  and messages are at most  $L$  bytes long. Although the universality of polynomial-evaluation-based hash functions is well established (as described in Section 2.2.7), the disparity between the field used to evaluate the polynomial and the group used for the encryption of the hash output means that this result is not inherited from other polynomial-based schemes. The outcome of this is that if the output of Poly1305 is encrypted by adding (modulo  $2^{128}$ ) a uniform random string then the resulting string is an information-theoretic message

## 6.2 The composition of ChaCha20 and Poly1305

---

authentication code (see [168] and [262]) and any adversary attempting to forge an authentication tag succeeds with probability at most  $\varepsilon$ .

### 6.2.2.3 The composition

The composition defined in Section 2.8 of draft-irtf-cfrg-chacha20-poly1305 [211] has three main parts, informally given below and more precisely described by  $E_k^0$  and  $D_k^0$  of Figure 6.1:

**Key derivation:** A one-time Poly1305 key,  $r$ , and pseudo-one-time-pad,  $s$ , are derived from  $k$  and  $N$  using the ChaCha20 block function with 0 as value of the block counter.

**Encryption:** The plaintext is encrypted using ChaCha20, with block  $i$  of the plaintext XORed with the output of ChaCha20 block function on input  $(k, i, N)$ .

**Tag generation:** Poly1305 is evaluated using the one time key  $r$ , which has certain bits set to zero. The input to Poly1305 is a message consisting of: associated data (padded with zeros to the next 128-bit block boundary), ciphertext (padded similarly), and a 128-bit block containing 64-bit representations of the length (in bytes) of both the associated data and the ciphertext. The pseudo-one-time-pad is then added (modulo  $2^{128}$ ) to the resulting digest.

### 6.2.3 Security model

The objective of this section is to demonstrate that the combination of ChaCha20 and Poly1305 described above is a secure authenticated encryption scheme, in the sense defined in Section 2.2.8. We will parametrise the adversary by the number of queries that are made to an encryption oracle and a decryption oracle ( $q_E$  and  $q_D$  queries respectively).

In this section, we use a slightly different definition of adversarial advantage than the one introduced in Section 2.2.8:

$$\mathbf{Adv}_{\text{CC\&Poly}}^{\text{aead}} = \left| \Pr[\mathcal{A}^{\text{CC\&Poly}} \rightarrow 1] - \Pr[\mathcal{A}^{\$, \perp} \rightarrow 1] \right| ,$$

## 6.2 The composition of ChaCha20 and Poly1305

---

where  $\$$  represents an oracle returning an appropriate number of random bits and  $\perp$  is an oracle that returns the distinguished value representing an invalid ciphertext on every input. These two definitions are well-known to be equivalent; noting that

$$\Pr[\text{AEAD}_{\text{AEAD}, \mathcal{A}}] = \Pr[b = 1] \Pr[\mathcal{A}^{\text{AEAD}} \rightarrow 1] + \Pr[b = 0] \Pr[\mathcal{A}^{\$, \perp} \rightarrow 0],$$

it is easy to see that

$$\begin{aligned} \left| 2 \Pr[\text{AEAD}_{\text{AEAD}, \mathcal{A}}] - 1 \right| &= \left| \Pr[\mathcal{A}^{\text{AEAD}} \rightarrow 1] + \Pr[\mathcal{A}^{\$, \perp} \rightarrow 0] - 1 \right| \\ &= \left| \Pr[\mathcal{A}^{\text{AEAD}} \rightarrow 1] + (1 - \Pr[\mathcal{A}^{\$, \perp} \rightarrow 1]) - 1 \right| \\ &= \left| \Pr[\mathcal{A}^{\text{AEAD}} \rightarrow 1] - \Pr[\mathcal{A}^{\$, \perp} \rightarrow 1] \right|. \end{aligned}$$

The proof proceeds via a series of games such that the AE-ENC and AE-DEC oracles in Game 0 realise CC&Poly and in Game 4 realise  $(\$, \perp)$ ; the AE-ENC and AE-DEC oracles in Game  $i$  are denoted  $E^i$  and  $D^i$  respectively.

All adversaries considered in this section will be restricted to nonce-respecting adversaries. This is common for nonce-based authenticated encryption schemes and means that an adversary will never ask encryption queries  $(N, A, P)$  and  $(N, A', P')$  for  $(A, P) \neq (A', P')$ . There is no restriction on the adversary's use of nonces for decryption queries. Without loss of generality, we only consider repeat- and redundancy-free adversaries (i.e. the input to an oracle is never repeated and the output from an E query is never input to the D oracle, or vice versa).

### 6.2.4 Our result

It is assumed in this security analysis that no pair  $(k, N')$  is ever repeated, where  $N'$  is the 96-bit nonce that is input to the ChaCha20 block function; this assumption is critical to the security of CC&Poly. The draft recognises that not all protocols will use 96-bit nonces and *'it is up to the protocol document to define how to transform the protocol nonce into a 96-bit nonce'* [211, Sect. 2.8]; one suggestion is that prepending a constant value could provide a way to expand a shorter nonce to 96 bits.

If an implementation permits both 96-bit nonces and shorter nonces and an adversary is able to predict how a short nonce will be expanded to 96 bits (for example, by guessing the value that will be prepended), then a nonce collision could be forced

## 6.2 The composition of ChaCha20 and Poly1305

---

by querying the encryption oracle using a short  $N$  and a 96-bit  $N'$  which is the expanded version of  $N$ .

The following theorem assumes that all nonces are 96 bits long and that no pair (key,nonce) is ever repeated to the encryption oracle; the protocol specification therefore must prevent nonce collisions of this form.

**Theorem 6.1.** *For every adversary  $\mathcal{A}$  against the AEAD security of CC&Poly, there is an adversary  $\mathcal{B}$  against the PRF security of the ChaCha20 block function, such that*

$$\mathbf{Adv}_{\text{CC\&Poly}}^{\text{aead}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{CC}}^{\text{prf}}(\mathcal{B}) + q_D \frac{8(\lceil L/128 \rceil)}{2^{106}},$$

where  $L = 128(\lceil \frac{L_A}{128} \rceil + \lceil \frac{L_C}{128} \rceil + 1)$ ,  $\mathcal{A}$  makes  $q_E$  encryption queries and  $q_D$  decryption queries, and  $\mathcal{B}$  makes at most  $(q_E + q_D)(\lceil L_C/512 \rceil + 1)$  queries.

*Proof.* The proof of this theorem proceeds via a series of games, specified in Figures 6.1 and 6.2. Game 0 defines a combined IND $\$$ -CPA and INT-CTXT game, with oracles that realise CC&Poly. The scheme specified in Game 4 clearly gives no adversary any advantage in either of the IND $\$$ -CPA and INT-CTXT games: the ciphertext blocks and tag are sampled independently of  $P$  and uniformly at random from  $\{0, 1\}^{512}$  (as they would be if generated by  $\$$ ) and it is impossible for an adversary to make a query to  $D_4$  that returns  $(N, A, P) \neq \perp$ .

The transitions between these games are justified as follows:

**Games 0 and 1** If an adversary is able to distinguish between these two games, then they can distinguish the ChaCha20 block function from a function chosen uniformly at random from the set of all functions with domain  $\{0, 1\}^{128}$  and range  $\{0, 1\}^{512}$ . The advantage gained by an adversary in this transition is therefore bounded above by  $\mathbf{Adv}_{\text{CC}}^{\text{prf}}(\mathcal{B})$ , where  $\mathcal{B}$  is an adversary that makes at most  $(q_E + q_D)(\lceil L_C/512 \rceil + 1)$  queries to its oracle in the PRF game.

**Games 1 and 2** These games are identical, on the condition that the inputs to  $\text{urf}$  in Game 1 never repeat. The inputs to  $\text{urf}$  are all of the form  $(i||N)$ ; for each query,  $N$  is constant, but  $i$  is never reused and no two encryption queries use the same value for  $N$  (as  $\mathcal{A}$  is nonce respecting and non-repeating), therefore the random variables in Games 1 and 2 are identically distributed.

## 6.2 The composition of ChaCha20 and Poly1305

---

**Games 2 and 3** These games are identical unless an adversary submits  $(N, A, C, \tau)$  to their decryption oracle and  $D_1$  returns  $(N, A, P) \neq \perp$ . However, for each query that an adversary makes, this happens with probability at most  $\varepsilon$  because  $\mathcal{A}$  is non-redundant and Poly is  $\varepsilon$ -almost  $\Delta$  universal, using results from [168], [34], and [262]. By a standard hybrid argument, the probability that an adversary making at most  $q_D$  queries to  $D$  successfully forges is at most  $\varepsilon q_D$ .

**Games 3 and 4** The random variables in these games are sampled in different orders, however the joint distributions are identical and therefore these games are identical.

A standard game-hopping argument allows the probability  $\Pr[\mathcal{A}^{G(i-1)} \rightarrow 1]$  to be bounded in terms of  $\Pr[\mathcal{A}^{G^i} \rightarrow 1]$ :

$$\begin{aligned} \Pr[\mathcal{A}^{G^0} \rightarrow 1] &\leq \Pr[\mathcal{A}^{G^1} \rightarrow 1] + \mathbf{Adv}_{\text{CC}}^{\text{prf}}(\mathcal{B}) , \\ \Pr[\mathcal{A}^{G^1} \rightarrow 1] &= \Pr[\mathcal{A}^{G^2} \rightarrow 1] , \\ \Pr[\mathcal{A}^{G^2} \rightarrow 1] &\leq \Pr[\mathcal{A}^{G^3} \rightarrow 1] + \varepsilon q_D , \\ \Pr[\mathcal{A}^{G^3} \rightarrow 1] &= \Pr[\mathcal{A}^{G^4} \rightarrow 1] = \Pr[\mathcal{A}^{\$, \perp} \rightarrow 1] , \end{aligned}$$

where  $\mathcal{B}$  makes at most  $(q_E + q_D)(\lceil L_C/512 \rceil + 1)$  queries.

Bernstein [34] demonstrates that Poly1305 is  $\varepsilon$ -almost  $\Delta$  universal for  $\varepsilon = \frac{8\lceil L/128 \rceil}{2^{106}}$  where  $L$  denotes the maximum bit length of messages and  $\Delta$  represents addition modulo  $2^{128}$ . For CC&Poly,  $L = 128(\lceil \frac{L_A}{128} \rceil + \lceil \frac{L_C}{128} \rceil + 1)$  as  $A$  and  $C$  are padded to 128-bit blocks and an extra 128 bits are added to encode the length of additional data and ciphertext.

Therefore it can be concluded that for every adversary  $\mathcal{A}$  there is an adversary  $\mathcal{B}$  against the PRF security of the ChaCha20 block function that makes  $(q_E + q_D)(\lceil L_C/512 \rceil + 1)$  queries in the PRF game such that:

$$\mathbf{Adv}_{\text{CC\&Poly}}^{\text{aead}}(\mathcal{A}) \leq \mathbf{Adv}_{\text{CC}}^{\text{prf}}(\mathcal{B}) + q_D \frac{8\lceil L/128 \rceil}{2^{106}} .$$

■

## 6.2 The composition of ChaCha20 and Poly1305

<p><u>Game <math>i</math>:</u>  <math>k \leftarrow_{\\$} \{0, 1\}^{256}</math>  <math>\text{urf} \leftarrow_{\\$} \text{Func}(\{0, 1\}^{128}, \{0, 1\}^{512})</math>  <math>b' \leftarrow_{\\$} \mathcal{A}^{\text{E}_k^i, \text{D}_k^i}</math>  Return <math>b'</math></p>	
<p><u>Game 0:</u></p> <p><math>\text{E}_k^0(N, A, P)</math>  <math>r    s \leftarrow \text{trunc}_{256}(\text{CC}_k(0    N))</math>  For <math>i = 1, \dots, p-1</math>:  <math>Z_i \leftarrow \text{CC}_k(i    N)</math>  <math>C_i \leftarrow Z_i \oplus P_i</math>  <math>Z_p^* \leftarrow \text{trunc}_{ P_p }(\text{CC}_k(p    N))</math>  <math>C_p \leftarrow Z_p^* \oplus P_p</math>  <math>\tau \leftarrow \text{Poly}_r(\overline{A}    \overline{C}    \text{len}_{A,C}) + s</math>  Return <math>(N, A, C, \tau)</math></p> <p><math>\text{D}_k^0(N, A, C, \tau)</math>  <math>r    s \leftarrow \text{trunc}_{256}(\text{CC}_k(0    N))</math>  <math>\tau' \leftarrow \text{Poly}_r(\overline{A}    \overline{C}    \text{len}_{A,C}) + s</math>  If <math>\tau \neq \tau'</math>  Return <math>\perp</math>  For <math>i = 1, \dots, p-1</math>:  <math>Z_i \leftarrow \text{CC}_k(i    N)</math>  <math>P_i \leftarrow Z_i \oplus C_i</math>  <math>Z_p^* \leftarrow \text{trunc}_{ C_p }(\text{CC}_k(p    N))</math>  <math>P_p \leftarrow Z_p^* \oplus C_p</math>  Return <math>(N, A, P)</math></p>	<p><u>Game 1:</u></p> <p><math>\text{E}_k^1(N, A, P)</math>  <math>r    s \leftarrow \text{trunc}_{256}(\text{urf}(0    N))</math>  For <math>i = 1, \dots, p-1</math>:  <math>Z_i \leftarrow \text{urf}(i    N)</math>  <math>C_i \leftarrow Z_i \oplus P_i</math>  <math>Z_p^* \leftarrow \text{trunc}_{ P_p }(\text{urf}(p    N))</math>  <math>C_p \leftarrow Z_p^* \oplus P_p</math>  <math>\tau \leftarrow \text{Poly}_r(\overline{A}    \overline{C}    \text{len}_{A,C}) + s</math>  Return <math>(N, A, C, \tau)</math></p> <p><math>\text{D}_k^1(N, A, C, \tau)</math>  <math>r    s \leftarrow \text{trunc}_{256}(\text{urf}(0    N))</math>  <math>\tau' \leftarrow \text{Poly}_r(\overline{A}    \overline{C}    \text{len}_{A,C}) + s</math>  If <math>\tau \neq \tau'</math>  Return <math>\perp</math>  For <math>i = 1, \dots, p-1</math>:  <math>Z_i \leftarrow \text{urf}(i    N)</math>  <math>P_i \leftarrow Z_i \oplus C_i</math>  <math>Z_p^* \leftarrow \text{trunc}_{ C_p }(\text{CC}_k(p    N))</math>  <math>P_p \leftarrow Z_p^* \oplus C_p</math>  Return <math>(N, A, P)</math></p>

Figure 6.1: Games 0 and 1 in the proof of Theorem 6.1.

<p><u>Game 2:</u></p> <p><math>E_k^2(N, A, P)</math></p> <p><math>r  s \leftarrow_s \{0, 1\}^{256}</math></p> <p>For <math>i = 1, \dots, p - 1</math>:</p> <p style="padding-left: 2em;"><math>Z_i \leftarrow_s \{0, 1\}^{512}</math></p> <p style="padding-left: 2em;"><math>C_i \leftarrow Z_i \oplus P_i</math></p> <p><math>Z_p^* \leftarrow_s \{0, 1\}^{ P_p }</math></p> <p><math>C_p \leftarrow Z_p^* \oplus P_p</math></p> <p><math>\tau \leftarrow \text{Poly}_r(\overline{A}  \overline{C}  \text{len}_{A,C}) + s</math></p> <p>Return <math>(N, A, C, \tau)</math></p> <p><math>D_k^2(N, A, C, \tau)</math></p> <p><math>r  s \leftarrow_s \{0, 1\}^{256}</math></p> <p><math>\tau' \leftarrow \text{Poly}_r(\overline{A}  \overline{C}  \text{len}_{A,C}) + s</math></p> <p>If <math>\tau \neq \tau'</math></p> <p style="padding-left: 2em;">Return <math>\perp</math></p> <p>For <math>i = 1, \dots, p - 1</math>:</p> <p style="padding-left: 2em;"><math>Z_i \leftarrow_s \{0, 1\}^{512}</math></p> <p style="padding-left: 2em;"><math>P_i \leftarrow Z_i \oplus C_i</math></p> <p><math>Z_p^* \leftarrow_s \{0, 1\}^{ C_p }</math></p> <p><math>P_p \leftarrow Z_p^* \oplus C_p</math></p> <p>Return <math>(N, A, P)</math></p>	<p><u>Game 3:</u></p> <p><math>E_k^3(N, A, P)</math></p> <p><math>r  s \leftarrow_s \{0, 1\}^{256}</math></p> <p>For <math>i = 1, \dots, p - 1</math>:</p> <p style="padding-left: 2em;"><math>Z_i \leftarrow_s \{0, 1\}^{512}</math></p> <p style="padding-left: 2em;"><math>C_i \leftarrow Z_i \oplus P_i</math></p> <p><math>Z_p^* \leftarrow_s \{0, 1\}^{ P_p }</math></p> <p><math>C_p \leftarrow Z_p^* \oplus P_p</math></p> <p><math>\tau \leftarrow \text{Poly}_r(\overline{A}  \overline{C}  \text{len}_{A,C}) + s</math></p> <p>Return <math>(N, A, C, \tau)</math></p> <p><math>D_k^3(N, A, C, \tau)</math></p> <p>Return <math>\perp</math></p>	<p><u>Game 4:</u></p> <p><math>E_k^4(N, A, P)</math></p> <p><math>r  \tau \leftarrow_s \{0, 1\}^{256}</math></p> <p>For <math>i = 1, \dots, p - 1</math>:</p> <p style="padding-left: 2em;"><math>C_i \leftarrow_s \{0, 1\}^{512}</math></p> <p style="padding-left: 2em;"><math>Z_i \leftarrow C_i \oplus P_i</math></p> <p><math>C_p \leftarrow_s \{0, 1\}^{ P_p }</math></p> <p><math>Z_p^* \leftarrow P_p \oplus C_p</math></p> <p><math>s \leftarrow \tau - \text{Poly}_r(\overline{A}  \overline{C}  \text{len}_{A,C})</math></p> <p>Return <math>(N, A, C, \tau)</math></p> <p><math>D_k^4(N, A, C, \tau)</math></p> <p>Return <math>\perp</math></p>
---	---	--

Figure 6.2: Games 2 to 4 in the proof of Theorem 6.1.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

The standardisation of cryptography for low-cost ultra-high frequency (UHF) RFID tags has begun; these standards allow (tag, interrogator and mutual) authentication and secure tag-interrogator communications to be securely implemented. Passive UHF RFID tags pose some major challenges when deploying cryptography as they are very limited in terms of the space available in silicon and the power available for on-tag computation. By comparison the high frequency (HF) RFID tags found in public transport ticketing and NFC applications are positively luxurious.

The standardisation group ISO/IEC SC31/WG7 is working on a set of cryptographic suites to provide security to wireless devices including UHF RFID tags. These cryptographic suites are presented as independent parts to a single standard ISO/IEC 29167. Within this multi-part standard, 29167-15 is based around very simple operations and is intended to provide tag, interrogator, and mutual authentication.

The primary conclusion of this section is that ISO/IEC 29167-15 offers poor—in fact non-existent—security. While the scheme in 29167-15 can be compromised in many ways, we leverage algebraic cryptanalysis to provide an elegant and efficient attack, recovering the entire key after eavesdropping just four authentications.

A secondary, but arguably more far-reaching conclusion, is a warning that technically poor proposals can advance far into the standardisation process. There are many existing sound (and standardised) cryptographic designs for both HF and UHF RFID tags and it is hoped that this analysis deters standardisation bodies and product developers from adopting schemes that have received little technical scrutiny.

#### 6.3.1 The standardisation landscape for UHF RFID

To understand why the standardisation work examined in this section is underway, it is necessary to understand the role of other standards in this field. The commands that can be sent to a (standardised) UHF RFID tag are defined in two documents

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

	command	RFU	SenRep	IncRepLen
length	8	2	1	1
value	d5 <sub>x</sub>	00 <sub>b</sub>	0 <sub>b</sub> /1 <sub>b</sub>	0 <sub>b</sub> /1 <sub>b</sub>

	CSI	Length	Message	RN	CRC
length	8	12	<i>variable</i>	16	16
value	★	★	★	handle	CRC-16

Table 6.1: The format of an AUTHENTICATE command in Gen2v2.

that have been published<sup>1</sup> by EPCglobal, part of GS1. The dominant standard covering all current large-scale deployments is known as Gen2v1 [103] and the final update to this standard was published in 2008. In 2013 the Gen2v2 standard was published [104], extending the functionality of Gen2v1. The most significant and far-reaching additions are optional over-the-air commands that allow the deployment of security functionality.

Gen2v2 defines the over-the-air commands for UHF RFID but this is all that it does. For instance, a command AUTHENTICATE is defined and this can be used to develop a solution for tag, interrogator, or mutual authentication. However all the security commands in Gen2v2, including AUTHENTICATE, have been deliberately designed to be flexible and crypto-agnostic; they are completely independent of any specific cryptographic technology. As an example, the format of the AUTHENTICATE command is given in Table 6.1. The `handle` and `CRC-16` are part of the communication protocol while `SenRep` and `IncRepLen` are application options. The fields marked ★ are the most important for our purposes; their values are not defined by Gen2v2. The `CSI` field identifies the cryptographic algorithm/protocol and the `Length/Message` fields locate the cryptographic payload being carried by the command.

For the cryptographic technology itself we would likely turn to the usual sources. NIST standardises cryptographic technologies such as the AES [208]. Other cryptographic technologies have been standardised within ISO/IEC SC27 including some, such as Present [59, 143] and cryptoGPS [123, 141], which are explicitly targeted at constrained environments.

---

<sup>1</sup>The Gen2v1 specifications are also standardised within ISO/IEC 18000-63 with Gen2v2 standardisation underway as a revision.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

However there is an implementation gap between the over-the-air commands and the cryptographic primitives. For example, the `AUTHENTICATE` command says nothing about how to achieve tag authentication using, say, a challenge-response authentication protocol and in particular does not specify which algorithms might be supported on the tag or interrogator. Similarly, the AES standard (FIPS-197 [208]) does not tell us how to use the AES block cipher to perform tag authentication; instead FIPS-197 tells us how a 128-bit output is derived from a 128-bit input and a key.

It is the goal of the work in ISO/IEC SC31/WG7, therefore, to provide a mapping between the cryptographic primitive and the generic over-the-air command; that is, to fill in the information marked `*` in the command above. This mapping is referred to as a cryptographic suite and the ISO/IEC 29167 standard consists of several parts, each describing a cryptographic suite and a solution. If one wishes to perform tag authentication using AES-128 then ISO/IEC 29167-10 is the cryptographic suite of interest. For tag authentication with Present-80, Grain-128a or cryptoGPS, parts 29167-11 [143], 29167-13 [139], and 29167-17 [141] are, respectively, the ones to use.

Many of the cryptographic suites in ISO/IEC 29167 are built on trusted or previously standardised primitives. However some of the cryptographic suites, most notably 29167-15, have been built around new, immature, or weak proposals. Unfortunately, the ISO/IEC voting structure is such that even a technically poor proposal can advance far through the standardisation process. Indeed, the long-term outcome for the final version of 29167-15 is not currently clear: one of the more likely outcomes is that it becomes a technical specification. This would temporarily halt work on the standard and provide the opportunity for public comment; after three years technical standards are either abandoned or re-introduced to the standards process.

#### 6.3.2 The first version of ISO/IEC 29167-15

One reason for the longevity of ISO/IEC 29167-15 is that patches have been applied throughout the voting process. All variants of ISO/IEC 29167-15 propose mechanisms for tag, interrogator, and mutual authentication. These mechanisms are simple and built around the supposed difficulty of analysing a combination of bitwise xor and integer addition. The first proposals were even simpler; see Table 6.2.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

Interrogator (secret key PSK)	Tag (secret key PSK)
Choose RnInt	
	$\xrightarrow{\text{RnInt}}$
	$\xleftarrow{\text{RnTag}}$
$\text{SK} = \text{PSK} \oplus \text{RnInt} \oplus \text{RnTag}$	Choose RnTag
Choose ChInt	$\text{SK} = \text{PSK} \oplus \text{RnInt} \oplus \text{RnTag}$
$\text{A} = \text{SK} \oplus \text{ChInt}$	
	$\xrightarrow{\text{A}}$
	$\xleftarrow{\text{B}}$
$\text{T} = \text{SK} \oplus \text{B}$	$\text{ChInt} = \text{SK} \oplus \text{A}$
$\text{REVERSE}(\text{T}) \stackrel{?}{=} \text{ChInt}$	$\text{RChInt} = \text{REVERSE}(\text{ChInt})$
	$\text{B} = \text{SK} \oplus \text{RChInt}$

Table 6.2: The first version of the tag authentication scheme with the notation used in the original document. All variables are 64 bits long.

There are many problems with the proposal in Table 6.2 but the most pressing is that there is no security. The sole use of a single operator (in this case bitwise xor) gives a differential attack. By eavesdropping an attacker recovers  $\text{RnInt}$ ,  $\text{RnTag}$ ,  $\text{A}$ , and  $\text{B}$ . The adversary can then make a fake tag that fools a legitimate reader without knowing the secret key  $\text{PSK}$ . The attack is outlined in Table 6.3 where we denote the variables in a subsequent run of the protocol with  $*$ .

To confirm that the fake tag is always accepted as genuine we observe that

$$\begin{aligned}
 \text{REVERSE}(\text{T}^*) &= \text{REVERSE}(\text{SK}^* \oplus \text{B} \oplus \text{REVERSE}(\text{X}) \oplus \Delta_{SK}) \\
 &= \text{REVERSE}(\text{SK} \oplus \text{B}) \oplus \text{X} = \text{REVERSE}(\text{SK} \oplus (\text{SK} \oplus \text{RChInt})) \oplus \text{X} \\
 &= \text{ChInt} \oplus \text{X} = \text{ChInt} \oplus \Delta_A \oplus \Delta_{SK} \\
 &= (\text{SK} \oplus \text{A}) \oplus (\text{A} \oplus \text{A}^*) \oplus (\text{SK} \oplus \text{SK}^*) = \text{A}^* \oplus \text{SK}^* = \text{ChInt}^* .
 \end{aligned}$$

A tag can be cloned after eavesdropping one legitimate authentication.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

Interrogator (secret key PSK)	Fake Tag
Choose RnInt* $SK^* = PSK \oplus RnInt^* \oplus RnTag^*$  Choose ChInt* $A^* = SK^* \oplus ChInt^*$  $T^* = SK^* \oplus B^*$ $REVERSE(T^*) \stackrel{?}{=} ChInt^*$	Choose RnTag* $SK^*$ is unknown $\Delta_I = RnInt \oplus RnInt^*$ $\Delta_T = RnTag \oplus RnTag^*$ Save $\Delta_{SK} = \Delta_I \oplus \Delta_T$  $A \oplus A^* = \Delta_A$ $X = \Delta_A \oplus \Delta_{SK}$ $B^* = B \oplus REVERSE(X) \oplus \Delta_{SK}$
$\xrightarrow{RnInt^*}$ $\xleftarrow{RnTag^*}$  $\xrightarrow{A^*}$ $\xleftarrow{B^*}$	

Table 6.3: Fooling a legitimate reader during tag authentication. The attacker has eavesdropped on one run of the protocol in Table 6.2. The (changing) parameters in this second run are indicated using \*.

#### 6.3.3 The working draft of ISO/IEC 29167-15

After this inauspicious start a variant was formally proposed as a working draft (WD). The tag authentication protocol is illustrated in Table 6.4. Interrogator authentication is provided by reversing the roles of the two participants while mutual authentication is derived by interleaving two sessions that establish tag and interrogator authentication.

We now show that the key can be recovered in a passive attack with high reliability. Indeed, suppose an attacker intercepts SRN and SORN from a legitimate authentication session. We then have that

$$SRN \oplus SORN = (RN + 0x55 \dots 55) \oplus RN .$$

The least significant bit of  $SRN \oplus SORN$  is always set to 1 and further analysis of  $SRN \oplus SORN$  is easy to make.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

Interrogator (secret key PSK)	Tag (secret key PSK)
Choose RN $\text{SRN} = \text{RN} \oplus \text{PSK} \quad \xrightarrow{\text{SRN}} \quad \text{RN} = \text{SRN} \oplus \text{PSK}$ $\text{A} = \text{RN} + 0\text{x}55 \cdots 55$ $\text{B} = \text{SORN} \oplus \text{PSK} \quad \xleftarrow{\text{SORN}} \quad \text{SORN} = \text{A} \oplus \text{PSK}$ $\text{B} \stackrel{?}{=} \text{RN} + 0\text{x}55 \cdots 55$	

Table 6.4: The first formal proposal of the ISO/IEC 29167-15 tag authentication scheme. For clarity the notation is slightly adapted from that used in documentation. The shared secret key PSK and all intermediate values are 64 bits long,  $\oplus$  denotes bitwise xor, and  $+$  denotes integer addition modulo  $2^{64}$ .

For instance, the  $2^{32}$  values of RN for which  $\text{RN} \wedge 0\text{x}55 \cdots 55 = 0\text{x}00 \cdots 00$  will give  $\text{SRN} \oplus \text{SORN} = 0\text{x}55 \cdots 55$  and so observations based on the distribution of  $\text{SRN} \oplus \text{SORN}$  can be used to recover RN and, via SRN, the shared secret key PSK.

For an alternative approach, we simplify the notation by setting  $\text{Z} = \text{SRN} \oplus \text{SORN}$ ,  $\text{R} = \text{RN}$ , and  $\text{C} = 0\text{x}55 \cdots 55$  and so  $\text{SRN} \oplus \text{SORN} = \text{Z} = \text{R} \oplus (\text{R} + \text{C})$ . Considering this equation bit-by-bit gives, for bit position  $j$  with  $j \geq 0$ ,

$$\text{Z}_j = \text{R}_j \oplus ((\text{R}_j + \text{C}_j + \text{T}_{j-1}) \bmod 2) = \text{C}_j \oplus \text{T}_{j-1}, = \text{C}_j \oplus \text{T}_{j-1},$$

where  $\text{T}_{j-1}$  denotes the carry given at bit  $j-1$  generated within the integer addition  $\text{R} + \text{C}$ . Setting  $\text{T}_{-1} = 0$ , the carry bit  $\text{T}_j$  for  $j \geq 0$  is computed as

$$\begin{aligned} \text{T}_j &= \text{MAJ}(\text{R}_j, \text{C}_j, \text{T}_{j-1}) = (\text{R}_j \wedge \text{C}_j) \oplus (\text{R}_j \wedge \text{T}_{j-1}) \oplus (\text{C}_j \wedge \text{T}_{j-1}) \\ &= (\text{R}_j \wedge (\text{C}_j \oplus \text{T}_{j-1})) \oplus (\text{C}_j \wedge \text{T}_{j-1}), \end{aligned}$$

where MAJ denotes the majority function. Hence, for  $j \geq 0$ ,

$$\begin{aligned} \text{Z}_{j+1} &= \text{C}_{j+1} \oplus (\text{R}_j \wedge (\text{C}_j \oplus \text{T}_{j-1})) \oplus (\text{C}_j \wedge \text{T}_{j-1}) \\ &= \text{C}_{j+1} \oplus (\text{R}_j \wedge \text{S}_j) \oplus (\text{C}_j \wedge (\text{Z}_j \oplus \text{C}_j)) \\ &= \text{C}_{j+1} \oplus (\text{R}_j \wedge \text{S}_j) \oplus (\text{C}_j \wedge (\text{Z}_j \oplus 1)). \end{aligned}$$

This means that, for  $j \geq 0$ , we have  $\text{R}_j \wedge \text{Z}_j = \text{Z}_{j+1} \oplus \text{C}_{j+1} \oplus (\text{C}_j \wedge (\text{Z}_j \oplus 1))$ , which we write, setting  $\text{V}_j = \text{R}_j \wedge \text{Z}_j$ , as

$$\text{V}_j = \text{Z}_{j+1} \oplus \text{C}_{j+1} \oplus (\text{C}_j \wedge (\text{Z}_j \oplus 1)) \text{ for } j \geq 0.$$

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

Looking at the expression for  $V_j$  we see that it consists entirely of arguments from  $Z$ , which is available to an eavesdropper, and  $C$  which is fixed and known. Thus if  $Z_j = 1$  for bit  $j$ , which we expect half the time, then we can compute  $R_j$  directly and the corresponding bit of the shared secret  $PSK$  is given by

$$PSK_j = R_j \oplus SRN_j .$$

Each bit of  $PSK_j$ , for  $0 \leq j \leq 62$  can be recovered and we expect to be able to recover all but the most significant bit of  $PSK_j$  with two intercepted authentications. The work-effort is negligible. Note that this gives us two possible values for the full 64-bit shared secret  $PSK$ . However these two keys are equivalent, that is they behave identically in the authentication protocol, and so they can both be used to impersonate a tag.

#### 6.3.4 The first committee draft of ISO/IEC 29167-15

Early versions of ISO/IEC 29167-15 (several new work item proposals and a working draft) were clearly weak and offered little promise. However, sufficiently many national bodies abstained or voted positively at each round of voting that the scheme moved forward towards standardisation anyway. Once we arrive at a committee draft (CD) the document should, in theory, be technically mature: subsequent stages of the process, namely draft international standard (DIS) and final draft international standard (FDIS), provide little opportunity for technical modification before publication. Yet as we will show in this section, even the latest versions of ISO/IEC 29167-15 are far from being technically mature and are, in fact, completely insecure.

To repair the weaknesses in the first formal submission to ISO/IEC WD 29167-15 a patched version was briefly proposed; see Table 6.5.

##### 6.3.4.1 Conventional observations

Again, we can immediately see that the least significant bit of  $SRNi \oplus SORNi$  is always set to 0. It is easy to find other faults and we simplify the notation by setting  $R = RNi$ ,  $C = 0x55 \dots 55$ , and  $S = SRNi$ , with  $SO = SORNi$ .

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

Interrogator (secret key PSK)	Tag (secret key PSK)
Choose $R_{Ni}$	
$SR_{Ni} = (R_{Ni} + C) \oplus PSK$	$R_{Ni} = (SR_{Ni} \oplus PSK) - C$
$SOR_{Ni} \oplus R_{Ni} \stackrel{?}{=} PSK + C$	$SOR_{Ni} = (PSK + C) \oplus R_{Ni}$

Table 6.5: A ‘patched’ version of the ISO/IEC 29167-15 tag authentication scheme from which the final committee draft version is derived. All variables are 64 bits long and  $C = 0x55 \dots 55$ .

This gives us  $S = (R + C) \oplus PSK$  and  $S0 = (PSK + C) \oplus R$ , so

$$S \oplus PSK = ((PSK + C) \oplus S0) + C . \tag{6.1}$$

We can use Equation 6.1 as a distinguisher to check if a possible value for the key PSK is a correct candidate. This can be done in several ways, but we illustrate a byte-by-byte approach, first considering the least significant byte of PSK.

Suppose  $p$ ,  $s$ ,  $so$  are the least significant bytes of PSK,  $S$ , and  $S0$  respectively. Then any  $x$  satisfying  $s \oplus x = ((x + 0x55) \oplus so) + 0x55$  is a good candidate for  $p$ . After eight to sixteen runs, one value should be predicted with close to 100% reliability and the least significant byte of the key is recovered. In parallel, we can process other bytes of PSK in the same manner.

There is a slight complication due to the possibility of a carry from one byte to another in the integer addition; at the same time the most significant bit of each byte might also need some attention. However, closer analysis when using particular values  $S$  and  $S0$  can be used to avoid significant carry propagation. This allows us to filter incorrect values and the few key candidates that remain can be tested against the tag to find the right one.

Passively eavesdropping on eight to sixteen authentication runs appears to be sufficient to recover each byte of the key PSK with good reliability. The work effort is negligible since all bytes can be treated in parallel.

## 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

### 6.3.4.2 Algebraic cryptanalysis

The scheme in ISO/IEC 29167-15 uses a set of very simple operations and algebraic cryptanalysis proves to be very effective.

Let  $n$  be the size of all variables in the protocol; in our case we have  $n = 64$ . We will assume that an attacker can eavesdrop on  $k$  runs of a uni-directional (tag or interrogator) authentication protocol. Since the mutual authentication protocol consists of two interleaved runs of a uni-directional protocol, observing  $k$  runs of the mutual authentication protocol will give identical results to  $2k$  runs of the uni-directional protocol. Without loss of generality we have implemented the attack on the protocol for uni-directional (tag or interrogator) authentication.

Denote the value of  $\text{SRNi}$  on the  $t^{\text{th}}$  run of the protocol by  $\text{SRNi}^t$ , similarly for  $\text{SORNi}$  and  $\text{RNi}$ . We use variables  $\mathbf{p}_i$ ,  $\mathbf{s}_{t,i}$ ,  $\mathbf{r}_{t,i}$ ,  $\mathbf{so}_{t,i}$ ,  $\mathbf{a}_{t,i}$ , and  $\mathbf{b}_{t,i}$  for  $0 \leq i < n$  and  $0 \leq t < k$  assuming that all strings are written using big-endian convention, i.e.  $\text{PSK} = \mathbf{p}_{n-1} \dots \mathbf{p}_1 \mathbf{p}_0$ . The equations used to represent the scheme will be defined over  $\mathbb{F}_2$ , using  $+$  to denote addition and  $*$  to denote multiplication.

We represent the computation of  $\text{SRNi}^t$  as

$$\mathbf{s}_{t,i} = \mathbf{p}_i + \mathbf{r}_{t,i} + \mathbf{a}_{t,i} + \mathbf{c}_i ,$$

for  $0 \leq i < n$ ,  $0 \leq t < k$ , where  $\mathbf{a}_{t,i}$  is the carry bit during the modular addition of  $\text{RNi}^t$  and  $\mathbf{C}$ . This gives, for  $0 \leq i < n$ ,

$$\begin{aligned} \mathbf{a}_{t,0} &= 0 \\ \mathbf{a}_{t,i} &= \text{MAJ}(\mathbf{r}_{t,(i-1)}, \mathbf{c}_{i-1}, \mathbf{a}_{t,(i-1)}) \\ &= \mathbf{r}_{t,(i-1)} * \mathbf{c}_{i-1} + \mathbf{r}_{t,(i-1)} * \mathbf{a}_{t,(i-1)} + \mathbf{a}_{t,(i-1)} * \mathbf{c}_{i-1} . \end{aligned}$$

Similarly the computation of  $\text{SORNi}^t$  can be represented (with the same indices) as

$$\begin{aligned} \mathbf{b}_{t,0} &= 0 , \\ \mathbf{so}_{t,i} &= \mathbf{p}_i + \mathbf{r}_{t,i} + \mathbf{b}_{t,i} + \mathbf{c}_i , \\ \mathbf{b}_{t,i} &= \mathbf{p}_{i-1} * \mathbf{c}_{i-1} + \mathbf{p}_{i-1} * \mathbf{b}_{t,(i-1)} + \mathbf{b}_{t,(i-1)} * \mathbf{c}_{i-1} , \end{aligned}$$

where the variable  $\mathbf{b}$  denotes the carry bits.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

Finally, we define equations to represent the observed values of  $\text{SRNi}^t$  and  $\text{SORNi}^t$  and the specified value of  $\mathbf{C}$ ;  $\mathbf{s}_{t,i} = \text{SRNi}_i^t$ ,  $\mathbf{so}_{t,i} = \text{SORNi}_i^t$ , and  $\mathbf{c}_i = \mathbf{C}_i$ . Our system is presented as polynomials over the finite field  $\mathbb{F}_2$ , i.e. all variables and coefficients take values in  $\{0, 1\}$ . We therefore include the equations of the form  $x^2 = x$  for every variable  $x$ .

The complete set of equations can be summarised as follows:

$$\left\{ \begin{array}{lll} \mathbf{s}_{t,i} = \mathbf{p}_i + \mathbf{r}_{t,i} + \mathbf{a}_{t,i} + \mathbf{c}_i & 0 \leq i < n & 0 \leq t < k \\ \mathbf{a}_{t,(i+1)} = \mathbf{r}_{t,i} * \mathbf{c}_i + \mathbf{r}_{t,i} * \mathbf{a}_{t,i} + \mathbf{a}_{t,i} * \mathbf{c}_i & 0 \leq i < n - 1 & 0 \leq t < k \\ \mathbf{a}_{t,0} = 0 & 0 \leq t < k & \\ \mathbf{so}_{t,i} = \mathbf{p}_i + \mathbf{r}_{t,i} + \mathbf{b}_{t,i} + \mathbf{c}_i & 0 \leq i < n & 0 \leq t < k \\ \mathbf{b}_{t,(i+1)} = \mathbf{p}_i * \mathbf{c}_i + \mathbf{p}_i * \mathbf{b}_{t,i} + \mathbf{b}_{t,i} * \mathbf{c}_i & 0 \leq i < n - 1 & 0 \leq t < k \\ \mathbf{b}_{t,0} = 0 & 0 \leq t < k & \\ \mathbf{s}_{t,i} = \text{SRNi}_i^t & 0 \leq i < n & 0 \leq t < k \\ \mathbf{so}_{t,i} = \text{SORNi}_i^t & 0 \leq i < n & 0 \leq t < k \\ \mathbf{c}_i = \mathbf{C}_i & 0 \leq i < n & \\ x^2 = x & \text{for all } x . & \end{array} \right.$$

This system of polynomial equations includes  $5nk + 2n$  variables and  $11nk + 3n$  equations of degree at most two. Of course this system can be greatly simplified, by substituting the variables that have a fixed value (e.g.  $\mathbf{c}_i, \mathbf{a}_{t,0}, \mathbf{b}_{t,0}$ ), as well as the ones observed in the protocol runs ( $\mathbf{s}_{t,i}$  and  $\mathbf{so}_{t,i}$ ). This reduces the number of variables to  $(3n - 2)k + n$ , and the number of equations to  $(7n - 4)k + n$ . For the parameter values of relevance to ISO/IEC 29167-15, the entire system will consist therefore of  $444k + 64$  equations in  $190k + 64$  variables, which can be constructed for the very small values of  $k$  that are required to recover the key. We use Gröbner bases algorithms to solve this system [77].

The average number of key bits recovered and the average time required to solve the system of equations are given in the following table and illustrated in Figure 6.3.

Number of protocol runs ( $k$ )	1	2	3	4
Average number of key bits recovered	19.7	51.1	59.4	61.7
Average run time (s)	8	83	113	255

The attack was implemented on SageMathCloud [257] and timed using Python's `timeit` function; any set-up time is assumed to be pre-computed or amortised over many protocol runs.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

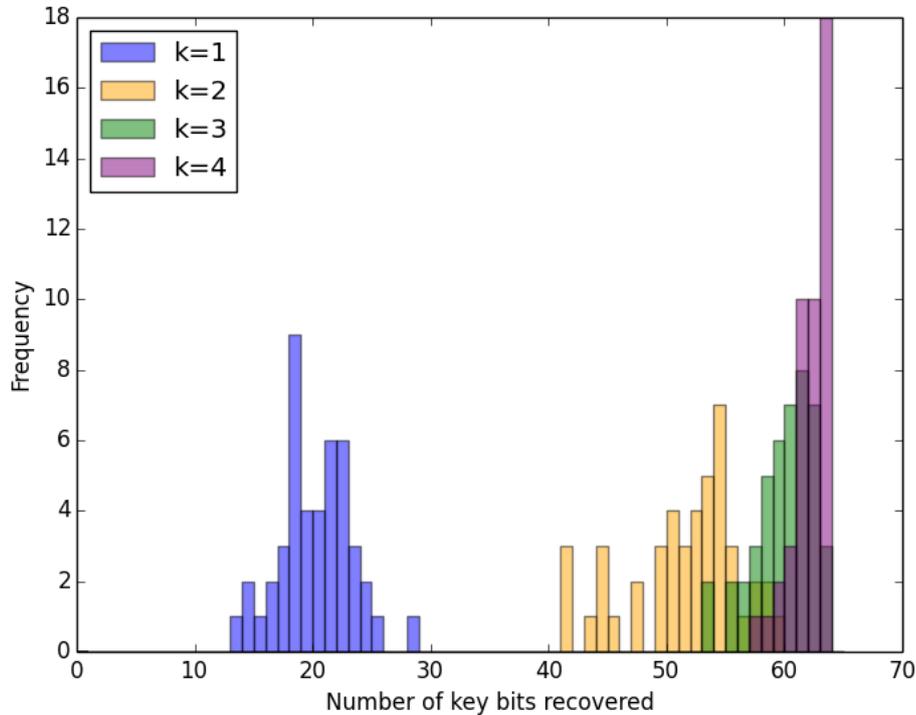


Figure 6.3: Results from experiments against the first committee draft of ISO/IEC 29167-15. The number of protocol runs is given by  $k$  and each set of experiments on  $k$  authentications was repeated 45 independent times.

The values  $(PSK, RNi^1)$  were chosen randomly for each trial when  $k = 1$ . As the number of protocol runs was increased  $PSK$  remains unchanged but fresh random choices were used for  $RNi^2$ ,  $RNi^3$ , and  $RNi^4$  as would be expected in a real-life implementation.

Our experiments suggest that after witnessing four uni-directional runs of the protocol (or just two mutual authentication runs) the attacker would be able to recover 62 out of 64 bits of the secret key in around 84% of the time. While the entirety of the key can often be recovered, we conjecture that the ‘missing’ bits that occur from time-to-time are neutral bits; the values of these bits cannot be determined by that particular instance of the equation system. This is a feature of many cryptanalytic techniques and is often exhibited in the most significant bits of operations such as integer addition. While it may be interesting to provide an exact explanation of this phenomenon, it is not relevant to the essential message of our cryptanalysis.

## 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

### 6.3.5 The final committee draft of ISO/IEC 29167-15

The fourth iteration to be described in this chapter is the final committee draft of ISO/IEC 29167-15. The mutual authentication protocol is given in Table 6.6. As in previous versions of the scheme, tag and interrogator authentication are derived from the relevant halves of the full authentication protocol.

Several changes have been made to this latest proposal to complicate the task of the cryptanalyst. The use of the (unknown) bitwise rotation seems to prevent the attacker from aligning bits in the challenge and response in a trivial way. For example a single bit change in the challenge will change the Hamming weight of the candidate  $RN_i$  derived by the tag; this would result in different rotation amounts being used for the computation of the final response.

Despite these complications, it is straightforward to compromise the scheme. Our initial analysis suggested that even with the rotation operation, the scheme can be compromised using conventional cryptanalysis after intercepting around 32 unidirectional authentication runs. However, as demonstrated in the previous section, it is more elegant and efficient to use algebraic cryptanalysis against ISO/IEC 29167-15 scheme. This technique allows us again to recover the shared secret key PSK after eavesdropping on as few as four authentication runs.

#### 6.3.5.1 Algebraic cryptanalysis of ISO/IEC 29167-15 final committee draft

As before, we need to set up a system of multivariate polynomial equations. The system used to describe this latest scheme is similar to that of Section 6.3.4. However it is helpful to introduce some additional variables to take account of the rotation:  $m_{t,i}$  corresponds to  $PSK'_i$  in the  $t^{\text{th}}$  protocol run and  $n_{t,i}$  corresponds to  $RN_i'_i$  in the  $t^{\text{th}}$  protocol run. In truth these variables have been introduced to improve the exposition of the attack. It would be straightforward to work without them if there were significant advantage in doing so.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

<b>Interrogator</b> (secret key PSK)	<b>Tag</b> (secret key PSK)
<p style="text-align: center;">Choose RN<sub>i</sub></p> $\text{SRN}_i = (\text{RN}_i + \text{C}) \oplus \text{PSK} \xrightarrow{\text{SRN}_i} \text{RN}_i = (\text{SRN}_i \oplus \text{PSK}) - \text{C}$	$\text{RN}_i = (\text{SRN}_i \oplus \text{PSK}) - \text{C}$ $w_i = \text{HW}(\text{RN}_i)$ $\text{PSK}' = \text{PSK} \lll w_i$ $\text{RN}_i' = \text{RN}_i \lll w_i$ $\text{SORN}_i = (\text{PSK}' + \text{C}) \oplus \text{RN}_i'$ <p style="text-align: center;">Choose RN<sub>t</sub></p> $\text{SRN}_t = (\text{RN}_t + \text{C}) \oplus \text{PSK}$
$w_i = \text{HW}(\text{RN}_i) \xleftarrow[\text{SRN}_t]{\text{SORN}_i}$ $\text{PSK}' = \text{PSK} \lll w_i$ $\text{RN}_i' = \text{RN}_i \lll w_i$ $(\text{SORN}_i \oplus \text{RN}_i') \stackrel{?}{=} (\text{PSK}' + \text{C})$ <p style="text-align: center;">tag authenticated</p>	$\text{RN}_t = (\text{SRN}_t \oplus \text{PSK}) - \text{C}$ $w_t = \text{HW}(\text{RN}_t)$ $\text{PSK}' = \text{PSK} \lll w_t$ $\text{RN}_t' = \text{RN}_t \lll w_t$ $\text{SORN}_t = (\text{PSK}' + \text{C}) \oplus \text{RN}_t' \xrightarrow{\text{SORN}_t} w_t = \text{HW}(\text{RN}_t)$ $\text{PSK}' = \text{PSK} \lll w_t$ $\text{RN}_t' = \text{RN}_t \lll w_t$ $(\text{SORN}_t \oplus \text{RN}_t') \stackrel{?}{=} (\text{PSK}' + \text{C})$ <p style="text-align: center;">interrogator authenticated</p>

Table 6.6: Mutual authentication, as specified in the latest version of ISO/IEC 29167-15. The Hamming weight of A is denoted HW(A) while  $A \lll w$  denotes the left bitwise rotation of A by  $w$  bits.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

In this variant of the scheme we need to take account of the unknown rotation amount. The simplest way to do this is to guess the rotation amount, and to solve each set of equations that arise for each guess. To include this within the equation system we introduce an array `rot_guess` where `rot_guess[t]` is a guess for the Hamming weight of  $\text{RNi}^t$ .

The complete set of equations can be summarised as follows:

$$\left\{ \begin{array}{lll} \mathbf{s}_{t,i} = \mathbf{p}_i + \mathbf{r}_{t,i} + \mathbf{a}_{t,i} + \mathbf{c}_i & 0 \leq i < n & 0 \leq t < k \\ \mathbf{a}_{t,(i+1)} = \mathbf{r}_{t,i} * \mathbf{c}_i + \mathbf{r}_{t,i} * \mathbf{a}_{t,i} + \mathbf{a}_{t,i} * \mathbf{c}_i & 0 \leq i < n - 1 & 0 \leq t < k \\ \mathbf{a}_{t,0} = 0 & 0 \leq t < k & \\ \mathbf{m}_{t,i} = \mathbf{P}_{(i+\text{rot\_guess}[t]\%n)} & 0 \leq i < n & 0 \leq t < k \\ \mathbf{n}_{t,i} = \mathbf{r}_{t,(i+\text{rot\_guess}[t]\%n)} & 0 \leq i < n & 0 \leq t < k \\ \mathbf{so}_{t,i} = \mathbf{m}_{t,i} + \mathbf{n}_{t,i} + \mathbf{b}_{t,i} + \mathbf{c}_i & 0 \leq i < n & 0 \leq t < k \\ \mathbf{b}_{t,(i+1)} = \mathbf{m}_{t,i} * \mathbf{c}_i + \mathbf{m}_{t,i} * \mathbf{b}_{t,i} + \mathbf{b}_{t,i} * \mathbf{c}_i & 0 \leq i < n - 1 & 0 \leq t < k \\ \mathbf{b}_{t,0} = 0 & 0 \leq t < k & \\ \mathbf{s}_{t,i} = \text{SRNi}_i^t & 0 \leq i < n & 0 \leq t < k \\ \mathbf{so}_{t,i} = \text{SORNi}_i^t & 0 \leq i < n & 0 \leq t < k \\ \mathbf{c}_i = \mathbf{C}_i & 0 \leq i < n & \\ x^2 = x & \forall x & \end{array} \right.$$

This system of equations includes  $7nk + 2n$  variables and  $15nk + 3n$  equations of degree at most two. Again, this system can be greatly simplified by substituting fixed/known value variables, as well as redundant ones, to a system with  $(3n-2)k+n$  variables, and  $(7n-4)k+n$  equations. For the parameter values of relevance to ISO/IEC 29167-15, the entire system consists of  $444k + 64$  equations in  $190k + 64$  variables, which can be constructed for the small values of  $k$  that are required to recover the key.

#### 6.3.5.2 Guessing the rotation amount

The equation system depends on `rot_guess`, an array of guesses for the Hamming weight of  $\text{RNi}$ . We expect that a correct guess for the Hamming weight of  $\text{RNi}$  will yield a system of equations that is easily solved to reveal many key bits.

The values  $\text{RNi}$  are random and so assuming that they are generated uniformly the random variable  $\text{HW}(\text{RNi})$  will be distributed according to a binomial distribution with parameters  $(64, \frac{1}{2})$ . It is therefore straightforward to compute the probability that a randomly chosen  $\text{RNi}$  has a particular Hamming weight.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

Of particular relevance to our attack is the fact that a substantial fraction of all possible  $\text{RNi}$  have a Hamming weight lying within a small range:

$x$	32	33	34	35	36
$\Pr_{\text{RNi}}[\text{HW}(\text{RNi}) = x]$	0.10	0.10	0.09	0.08	0.06

This means that the probability  $\text{HW}(\text{RNi})$  lies in the interval  $[32 - \delta, 32 + \delta]$  is:

$\delta$	0	1	2	3	4
$\Pr_{\text{RNi}}[\text{HW}(\text{RNi}) \in [32 - \delta, 32 + \delta]]$	0.10	0.29	0.46	0.62	0.74

Now assume an attacker eavesdrops on one uni-directional authentication session. He could simply run the equation solving algorithm three times with the guesses of 31, 32, and 33 for the rotation amount. With a probability close to 30% one of these guesses would be correct. Alternatively, he could elect to further try the values 28, 29, 30, 34, 35, and 36 which would require nine runs of the equation solving algorithm. The probability that the eavesdropped session is covered by one of these nine guesses is close to 74%. This has been confirmed by experiments.

It turns out that the Gröbner basis algorithm provides a good method to verify whether the guessed rotation amount is correct. Empirically, it seems that selecting the wrong rotation makes the system inconsistent, i.e. there will be no valid solution, and this is quickly detected by the Gröbner basis algorithm. Of course it cannot be ruled out that cases exist when an incorrect guess of rotation results in a system for which solutions exist (corresponding to an incorrect key). However this does not appear to be common; in over 20 experiments no false solutions were found for any  $\delta \leq 5$ . Of course, even if they did occur, false alarms could easily be filtered out using further intercepted authentication attempts or even a forgery attempt.

#### 6.3.5.3 Results

The average number of key bits recovered and the average time required to solve the system of equations are given in the following table and illustrated in Figure 6.4.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

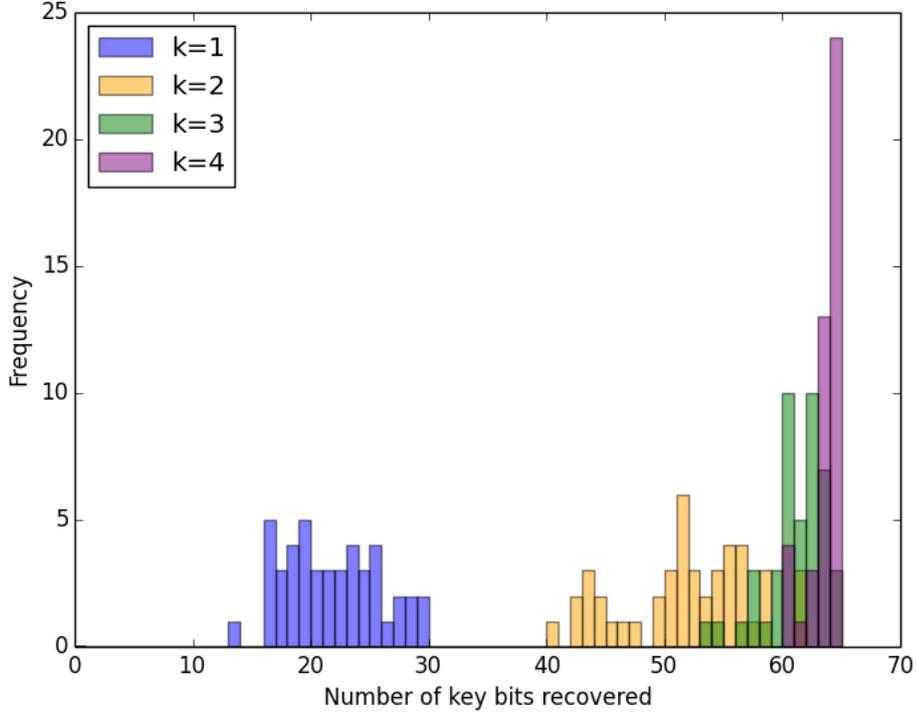


Figure 6.4: Results of experiments against the final committee draft of ISO/IEC 29167-15. The number of protocol runs is given by  $k$  and 45 random instances were generated. In each case, it is assumed that the rotation amount was guessed correctly.

These results assume a correct guess for each of the  $\text{HW}(\text{RNi}^j)$ , as discussed in the previous section.

Number of protocol runs ( $k$ )	1	2	3	4
Average number of key bits recovered	21.3	51.4	60.6	63.2
Average run time (s)	10	53	87	193

Again, the attack was implemented on SageMathCloud [257] and timed using the `timeit` function provided by Python; any set-up time is assumed to be pre-computed or amortised over many protocol runs. The values  $(\text{PSK}, \text{RNi}^1)$  were chosen randomly for each trial when  $k = 1$ . As the number of protocol runs was increased  $\text{PSK}$  remains unchanged but fresh random choices were used for  $\text{RNi}^2$ ,  $\text{RNi}^3$ , and  $\text{RNi}^4$  as would be expected in a real-life implementation.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

Witnessing four uni-directional runs, or two mutual authentication runs, of the scheme in Section 6.3.5 and guessing the rotation amount correctly gives a probability of approximately 85% for recovering all but two bits of the key. However different attack strategies are possible.

Note that the attack using a single observed run is much faster than the attack with four observed runs, although it would recover a smaller proportion of the secret key bits. Moreover, as discussed above, when attempting to solve the resulting system of equations it is straightforward to recognise when an incorrect rotation has been guessed. Thus an efficient way to perform the full attack is as follows: we repeat the single-observation attack ( $k = 1$ ) several times for each run ( $1 \leq t \leq 4$ ) of the protocol with different guesses for  $\text{HW}(\text{RNi}^t)$ . Once we have identified the correct rotation for all four runs of the protocol we would then be able to apply the four-run ( $k = 4$ ) attack thereby recovering almost all bits of the key with high probability.

There are numerous possible trade-offs related to this strategy. For example, repeating the single run attack for guesses in the range  $[30, 34]$  gives a  $\sim 46\%$  chance of finding the correct rotation. Doing this for eight observed protocol runs requires us to run the single-round attack at most 40 times to have a probability of  $\sim 54\%$  to recover the correct rotations for four runs of the protocol. By combining this with a second phase that attacks four runs ( $k = 4$ ) with correct rotation amounts, all but two bits of the key will be recovered with probability approximately  $0.54 \times 0.85 \approx 46\%$ . This is expected to take around 10 minutes to run on SageMathCloud.

Alternatively, the attacker could repeat the attack for each guess in the range  $[28, 36]$ , giving a 74% chance of guessing the rotation amounts correctly. Doing this for six observed protocol runs requires us to run the single-round attack at most 54 times and would give a probability of  $\sim 81\%$  to recover the rotations for four runs of the protocol. This attack would recover all except two bits of the key with probability approximately  $0.81 \times 0.85 \approx 69\%$  and take around 20 minutes to run on SageMathCloud.

#### 6.3.6 Other insecure variants of ISO/IEC 29167-15

With the hope of discouraging further patches to ISO/IEC 29167-15, we pro-actively anticipate some modifications that might be made with the hope of increasing security. In this section, we show that none of the obvious variants provide a significantly increased level of security.

One variant would be to increase the size of all parameters, using a secret key of size  $n = 64 + n'$  bits. The intention would be to increase the size of the equations systems since algebraic cryptanalytic schemes do not scale well. However this is particularly ineffective for the scheme of Table 6.5 since one can simply discard the  $n'$  high bits of the transmitted values and run precisely the same attack, recovering many of the lower 64 bits. We could then guess the value of the carry bit  $a_{j,64}$  which would allow the remaining bits to be attacked independently. We view this kind of attack as ‘slicing’ the problem and we will return to this below.

We can apply a similar technique to remove any advantage from increased parameters in the scheme of Table 6.6. The rotation  $r = \text{HW}(\text{RNi})$  will, with high probability, lie in an interval  $(\frac{n}{2} - \delta, \frac{n}{2} + \delta)$  for small  $\delta$ . This means we can consider a subset of equations consisting of  $\text{SRNi}_0^j \dots \text{SRNi}_{s-1}^j$  and  $\text{SORNi}_r^j \dots \text{SORNi}_{r+s-1}^j$  (and the corresponding  $p_i, r_{j,i}, a_{j,i}, b_{j,i}$ , etc.) for some value of the rotation  $r$ . As before, this attack requires us to guess the value of the rotations, and additionally we must guess the value of  $b_{j,x}$ ; this does not significantly affect the complexity of the attack. (In this case we do not need to guess  $a_{j,0}$ , however had we considered a subset of equations not including  $\text{SRNi}_0^j$  we would have needed to guess another carry bit.)

Figure 6.5 illustrates the results of implementing this attack. From two runs of the uni-directional authentication, the attacker guesses the values of four bits and, for the correct rotation amount, the number of key bits recovered when using a single 16-bit or 24-bit ‘slice’. One strategy that may improve the efficiency of our attack would be to first attack a small number of bits (via the slicing attack) for several possible guessed values of  $\text{HW}(\text{RNi}^j)$ . Then to carry out a full attack against all  $k$  protocol runs (simultaneously) once the correct rotation values have been identified. This way one could recover almost all the key bits without having to run the full attack many times.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

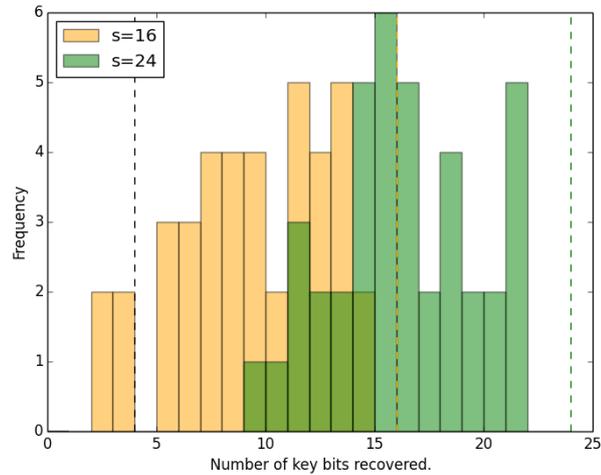


Figure 6.5: Results of the ‘slicing’ attack applied against the final committee draft of ISO/IEC 29167-15 with two runs of the uni-directional authentication scheme. The values of four bits were guessed and the black dashed line highlights this value. Dashed lines (green and yellow) show the maximum possible number of key bits that can be recovered, this restriction being set by the size of the ‘slice’ ( $s$ ).

An alternative modification to the protocol might be to change the value of the constant  $C$ . We implemented an attack assuming that two runs of the uni-directional authentication protocol were observed. We implemented a 16-bit ‘slicing’ attack and repeated the whole set of experiments twice (using fresh random choices of PSK and  $RN_i$ ) for each of 256 different values of  $C$ . The 256  $C$  values were built up as a single byte pattern repeated eight times. The number of key bits that were recovered in our attacks for different  $C$  are illustrated in Figure 6.6.

Zero is the only value for  $C$  that leaked no key bits. However  $0^{64}$  is a particularly bad choice for  $C$  since this makes  $SRN_i = SORN_i$  and forgeries are trivial; in this case no key bits were recovered because if  $C$  is zero then the protocol transcript is independent of the key. Every other choice of  $C$  leads to at least four key bits out of 16 being recovered. This suggests that changing the value of  $C$  is unlikely to improve the security of the proposed protocol.

### 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

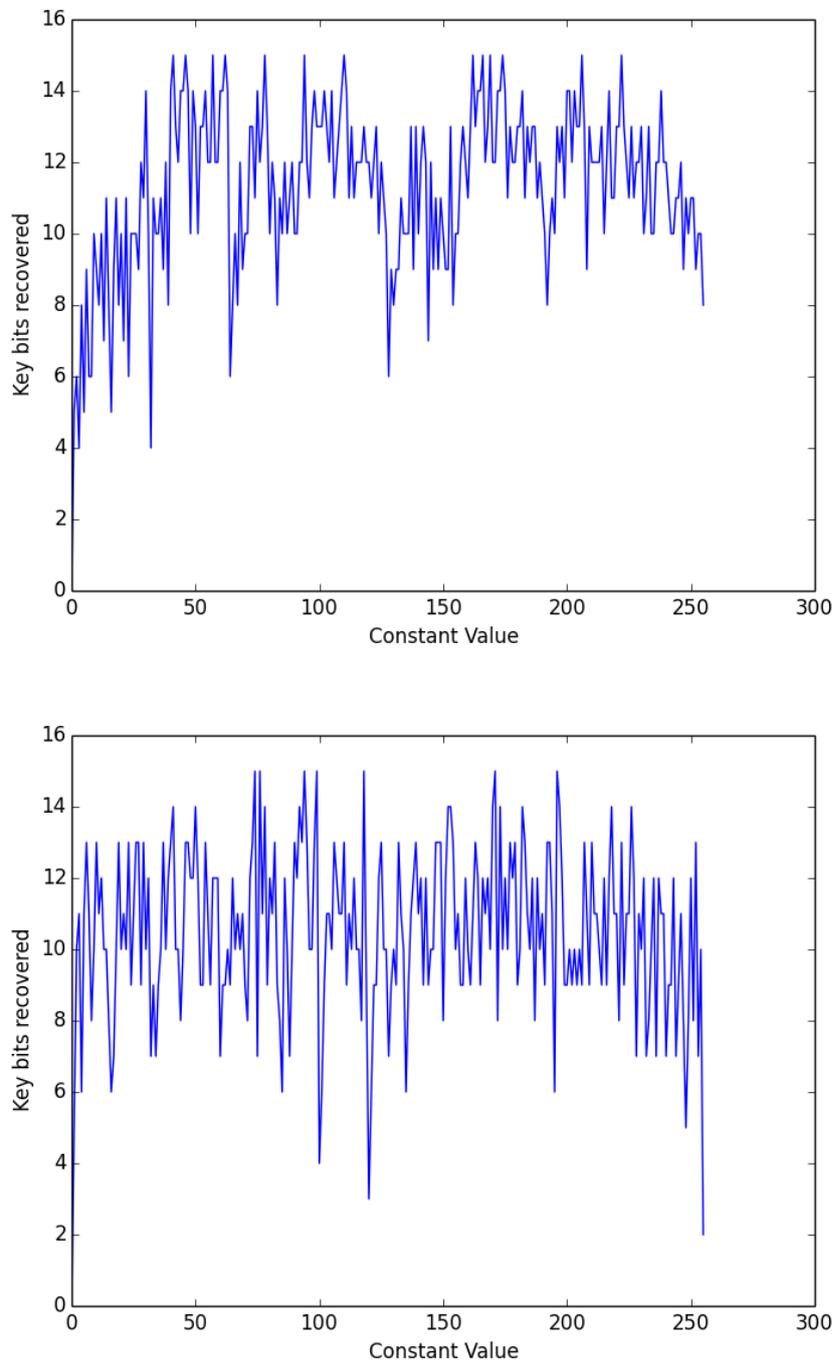


Figure 6.6: Results of implementing the attack against variants of ISO/IEC 29167-15 where the constant  $C$  moves through 256 possible values. Experiments were run twice, each using two runs of the uni-directional authentication scheme.

## 6.3 Algebraic cryptanalysis and ISO/IEC 29167-15

---

### 6.3.7 Results and discussion

Through this section we have seen several incremental versions of the work in ISO/IEC 29167-15 and observed that none offer any substantive security. We have also seen that future versions, if based on identical principles, are unlikely to provide additional security. These results are summarised here, with all attacks applicable to passive adversaries; most results in this section have been implemented.

Version	Net Result	# Authentications
Table 6.2	Tag cloning	1
Table 6.4	Key recovery	2
Table 6.5	Key recovery	8–16
Table 6.5	Key recovery	4
Table 6.6	Key recovery	4

Although we have concentrated on uni-directional authentication, mutual authentication simply consists of two interleaved versions of a tag and interrogator authentication. This means that all of our attacks will also apply to mutual authentication, but often with less effort since twice as much information is leaked during each protocol run.

The scheme in ISO/IEC 29167-15 is intrinsically weak due to the simple operations used by the tag and the interrogator; for all but the first variant the long term PSK key can be recovered. For completeness, we note that the latest draft of ISO/IEC WD 29167-15 also includes a method to provide a secure channel, but the encryption method is wholly insecure.

The project editors for 29167-15 motivate their use of the simplest operations by stating that this will result in a low-area solution. However this is misguided: the bulk of the area for an implementation comes from the cryptographic state which is governed by the size of the variables. So even though ISO/IEC 29167-15 uses simple operations it doesn't lead to a dramatic implementation advantage. More importantly, there are already very good cryptographic solutions for UHF RFID tags that provide good security; the AES is one option and those that prefer a bit more implementation agility might find that Present [143] or Grain-128a [1] provide different security/area/performance trade-offs.

## 6.4 Discussion

---

One goal of the work presented in this section was to demonstrate that ISO/IEC 29167-15 is broken and to emphasise the contributions that cryptographers can make to a variety of ISO/IEC initiatives [93]. A second, more important, goal was to stress that cryptography for RFID does not need to be bad cryptography. The state of the art is such that well-studied standardised schemes are available and these can be deployed in even the most demanding environments.

The code used to generate the data in this section can be found at <http://www.isg.rhul.ac.uk/~ccid/publications/iso-iec-29167-15.htm>.

## 6.4 Discussion

This chapter has shown two sides of the standardisation of cryptosystems. On the one hand, we have ChaCha20 and Poly1305: they are mature, well-established algorithms that have been combined into an AEAD construction in order to surface a more appropriate interface and that construction is (now) accompanied by a security proof. The other hand illustrates a much less optimistic picture: a novel construction is rushed through standardisation with little analysis, heavily tweaked, and kept alive by politics and voting patterns rather than technical merit. This pessimistic picture is not unique to ISO/IEC 29167-15; it is reminiscent of PLAID [93] and the HB+ proposals [154] (where a string of attacks [120, 182, 97, 121] alternated with new versions [12, 64, 203, 122]).

Many insecure RFID authentication protocols have been proposed. LMAP [218], M2AP [219] and EMAP [220] all somewhat resemble ISO/IEC 29167-15 as they also use integer additions, bitwise rotations, and exclusive-ors (ARX) for the basis of the construction; they too have been broken [183, 133, 184].

These ARX designs can be a good basis from which to build a secure primitive: they are (rightly) popular and featured prominently in the NIST SHA-3 initiative [213]. However these are typically multi-round algorithms, whereas the ultra-lightweight RFID authentication protocols have perhaps achieved just a ‘single round’ of computational complexity (if one can make the analogy). Fortunately, ChaCha20 is an example of an ARX-based scheme that is currently believed to be secure and, as the

## 6.4 Discussion

---

analysis contained in this chapter demonstrates, the novel combination of ChaCha20 and Poly1305 is also secure.

It is reassuring that, of the two schemes analysed in this chapter, it is the combination of ChaCha20 and Poly1305 that has seen widespread adoption. This perhaps reflects the confidence provided by choosing established algorithms and explicitly requesting a security analysis of the novel component, compared with the process of repeatedly refining and tweaking a poor initial design.

# Bibliography

---

- [1] Martin Ågren, Martin Hell, Thomas Johansson, and Willi Meier. Grain-128a: a new version of Grain-128 with optional authentication. *International Journal of Wireless and Mobile Computing*, 5(1):48–59, 2011.
- [2] Mohamed Ahmed Abdelraheem, Peter Beelen, Andrey Bogdanov, and Elmar Tischhauser. Twisted polynomials and forgery attacks on GCM. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 762–786. Springer, April 2015.
- [3] Mohamed Ahmed Abdelraheem, Andrey Bogdanov, and Elmar Tischhauser. Weak-key analysis of POET. Cryptology ePrint Archive, Report 2014/226, 2014. <http://eprint.iacr.org/2014/226>.
- [4] Farzaneh Abed, Scott Fluhrer, John Foley, Christian Forler, Eik List, Stefan Lucks, David McGrew, and Jakob Wenzel. The POET family of on-line authenticated encryption schemes. *Submission to the CAESAR competition*, 2014. <http://competitions.cr.yyp.to/round1/poetv101.pdf>.
- [5] Martin R. Albrecht, Pooya Farshim, Kenneth G. Paterson, and Gaven J. Watson. On cipher-dependent related-key attacks in the ideal-cipher model. In Antoine Joux, editor, *Fast Software Encryption – FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 128–145. Springer, February 2011.
- [6] Martin R. Albrecht, Kenneth G. Paterson, and Gaven J. Watson. Plaintext recovery attacks against SSH. In *2009 IEEE Symposium on Security and Privacy*, pages 16–26. IEEE Computer Society Press, May 2009.

## BIBLIOGRAPHY

---

- [7] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 83–107. Springer, April / May 2002.
- [8] Ross J. Anderson and Markus G. Kuhn. Low cost attacks on tamper resistant devices. In Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe, editors, *Security Protocols, 5th International Workshop, Paris, France, April 7-9, 1997, Proceedings*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1997.
- [9] Elena Andreeva, Andrey Bogdanov, Yevgeniy Dodis, Bart Mennink, and John P. Steinberger. On the indistinguishability of key-alternating ciphers. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 531–550. Springer, August 2013.
- [10] Elena Andreeva, Bart Mennink, and Bart Preneel. On the indistinguishability of the Grøstl hash function. In Juan A. Garay and Roberto De Prisco, editors, *SCN 10: 7th International Conference on Security in Communication Networks*, volume 6280 of *Lecture Notes in Computer Science*, pages 88–105. Springer, September 2010.
- [11] Kazumaro Aoki and Kan Yasuda. The security and performance of “GCM” when short multiplications are used instead. In Mirosław Kutylowski and Moti Yung, editors, *Information Security and Cryptology*, volume 7763 of *Lecture Notes in Computer Science*, pages 225–245. Springer Berlin Heidelberg, 2013.
- [12] Frederik Armknecht, Matthias Hamann, and Vasily Mikhalev. Lightweight authentication protocols on ultra-constrained RFIDs—myths and facts. In *Radio Frequency Identification: Security and Privacy Issues*, pages 1–18. Springer, 2014.
- [13] Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New features of latin dances: Analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *Fast Software Encryption – FSE 2008*, volume 5086 of *Lecture Notes in Computer Science*, pages 470–488. Springer, February 2008.

## BIBLIOGRAPHY

---

- [14] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and María Naya-Plasencia. Quark: A lightweight hash. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems – CHES 2010*, volume 6225 of *Lecture Notes in Computer Science*, pages 1–15. Springer, August 2010.
- [15] Manuel Barbosa and Pooya Farshim. The related-key analysis of Feistel constructions. In Carlos Cid and Christian Rechberger, editors, *Fast Software Encryption – FSE 2014*, volume 8540 of *Lecture Notes in Computer Science*, pages 265–284. Springer, March 2015.
- [16] Gregory V. Bard, Nicolas T. Courtois, and Chris Jefferson. Efficient methods for conversion and solution of sparse systems of low-degree multivariate polynomials over GF(2) via SAT-Solvers. Cryptology ePrint Archive, Report 2007/024, 2007. <http://eprint.iacr.org/2007/024>.
- [17] William C. Barker and Elaine Barker. Recommendation for the triple data encryption algorithm (TDEA) block cipher, Revision 1. NIST Special Publication 800-67, <http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf>, Jan 2012.
- [18] Mihir Bellare. Practice-oriented provable-security. In Eiji Okamoto, George Davida, and Masahiro Mambo, editors, *Information Security*, volume 1396 of *Lecture Notes in Computer Science*, pages 221–231. Springer Berlin Heidelberg, 1998.
- [19] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstanciatable random-oracle-model scheme for a hybrid-encryption problem. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188. Springer, May 2004.
- [20] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying hash functions for message authentication. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO’96*, volume 1109 of *Lecture Notes in Computer Science*, pages 1–15. Springer, August 1996.
- [21] Mihir Bellare and David Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In Tal Rabin, editor, *Advances*

## BIBLIOGRAPHY

---

- in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 666–684. Springer, August 2010.
- [22] Mihir Bellare, David Cash, and Rachel Miller. Cryptography secure against related-key attacks and tampering. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 486–503. Springer, December 2011.
- [23] Mihir Bellare, Anand Desai, Eric Jorjipii, and Phillip Rogaway. A concrete security treatment of symmetric encryption. In *38th Annual Symposium on Foundations of Computer Science*, pages 394–403. IEEE Computer Society Press, October 1997.
- [24] Mihir Bellare, Roch Guérin, and Phillip Rogaway. XOR MACs: New methods for message authentication using finite pseudorandom functions. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO’95*, volume 963 of *Lecture Notes in Computer Science*, pages 15–28. Springer, August 1995.
- [25] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO’94*, volume 839 of *Lecture Notes in Computer Science*, pages 341–358. Springer, August 1994.
- [26] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of the cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000.
- [27] Mihir Bellare and Tadayoshi Kohno. A theoretical treatment of related-key attacks: RKA-PRPs, RKA-PRFs, and applications. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, May 2003.
- [28] Mihir Bellare, Ted Krovetz, and Phillip Rogaway. Luby-Rackoff backwards: Increasing security by making block ciphers non-invertible. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT’98*, volume 1403 of *Lecture Notes in Computer Science*, pages 266–280. Springer, May / June 1998.
- [29] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In

## BIBLIOGRAPHY

---

- Tatsuaki Okamoto, editor, *Advances in Cryptology – ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer, December 2000.
- [30] Mihir Bellare, Krzysztof Pietrzak, and Phillip Rogaway. Improved security analyses for CBC MACs. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 527–545. Springer, August 2005.
- [31] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73. ACM Press, November 1993.
- [32] Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, May / June 2006.
- [33] E. R. Berlekamp. Factoring polynomials over large finite fields. *Math. Comp.*, 24:713–735, 1970.
- [34] Daniel J. Bernstein. The Poly1305-AES message-authentication code. In Henri Gilbert and Helena Handschuh, editors, *Fast Software Encryption – FSE 2005*, volume 3557 of *Lecture Notes in Computer Science*, pages 32–49. Springer, February 2005.
- [35] Daniel J. Bernstein. ChaCha, a variant of Salsa20, 2008. <http://cr.yp.to/papers.html#chacha>. Document ID: 4027b5256e17b9796842e6d0f68b0b5e.
- [36] Daniel J. Bernstein. The Salsa20 family of stream ciphers. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs*, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97. Springer Berlin Heidelberg, 2008.
- [37] Daniel J. Bernstein and Tanja Lange. Non-uniform cracks in the concrete: The power of free precomputation. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 321–340. Springer, December 2013.

## BIBLIOGRAPHY

---

- [38] Daniel J. Bernstein, Tanja Lange, and Peter Schwabe. The security impact of a new cryptographic library. In Alejandro Hevia and Gregory Neven, editors, *Progress in Cryptology - LATINCRYPT 2012: 2nd International Conference on Cryptology and Information Security in Latin America*, volume 7533 of *Lecture Notes in Computer Science*, pages 159–176. Springer, October 2012.
- [39] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. Cryptographic sponge functions. <http://sponge.noekeon.org/CSF-0.1.pdf>, 2011.
- [40] Guido Bertoni, Joan Daemen, Michaël Peeters, and Gilles Van Assche. The Keccak reference. <http://keccak.noekeon.org/Keccak-reference-3.0.pdf>, 2011.
- [41] Guido Bertoni, Joan Daemen, Michael Peeters, and Gilles Van Assche. On the indifferentiability of the sponge construction. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 181–197. Springer, April 2008.
- [42] Jürgen Bierbrauer, Thomas Johansson, Gregory Kabatianskii, and Ben Smeets. On families of hash functions via geometric codes and concatenation. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO’93*, volume 773 of *Lecture Notes in Computer Science*, pages 331–342. Springer, August 1994.
- [43] Eli Biham. New types of cryptanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, 1994.
- [44] Eli Biham. New types of cryptoanalytic attacks using related keys (extended abstract). In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT’93*, volume 765 of *Lecture Notes in Computer Science*, pages 398–409. Springer, May 1994.
- [45] Alex Biryukov. Feistel cipher. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 455–455. Springer US, 2011.
- [46] Alex Biryukov. Weak keys. In Henk C. A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 1366–1367. Springer, 2011.

## BIBLIOGRAPHY

---

- [47] Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, December 2009.
- [48] Alex Biryukov, Dmitry Khovratovich, and Ivica Nikolic. Distinguisher and related-key attack on the full AES-256. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 231–249. Springer, August 2009.
- [49] Alex Biryukov and David Wagner. Slide attacks. In Lars R. Knudsen, editor, *Fast Software Encryption – FSE’99*, volume 1636 of *Lecture Notes in Computer Science*, pages 245–259. Springer, March 1999.
- [50] Alex Biryukov and David Wagner. Advanced slide attacks. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 589–606. Springer, May 2000.
- [51] John Black. The ideal-cipher model, revisited: An uninstantiable blockcipher-based hash function. In Matthew J. B. Robshaw, editor, *Fast Software Encryption – FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 328–340. Springer, March 2006.
- [52] John Black and Martin Cochran. MAC reforgeability. In Orr Dunkelman, editor, *Fast Software Encryption – FSE 2009*, volume 5665 of *Lecture Notes in Computer Science*, pages 345–362. Springer, February 2009.
- [53] John Black, Shai Halevi, Hugo Krawczyk, Ted Krovetz, and Phillip Rogaway. UMAC: Fast and secure message authentication. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO’99*, volume 1666 of *Lecture Notes in Computer Science*, pages 216–233. Springer, August 1999.
- [54] John Black and Phillip Rogaway. CBC MACs for arbitrary-length messages: The three-key constructions. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 197–215. Springer, August 2000.
- [55] John Black and Phillip Rogaway. A block-cipher mode of operation for parallelizable message authentication. In Lars R. Knudsen, editor, *Advances in*

## BIBLIOGRAPHY

---

- Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 384–397. Springer, April / May 2002.
- [56] John Black, Phillip Rogaway, and Thomas Shrimpton. Black-box analysis of the block-cipher-based hash-function constructions from PGV. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 320–335. Springer, August 2002.
- [57] Martin Boesgaard, Thomas Christensen, and Erik Zenner. Badger - a fast and provably secure MAC. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *ACNS 05: 3rd International Conference on Applied Cryptography and Network Security*, volume 3531 of *Lecture Notes in Computer Science*, pages 176–191. Springer, June 2005.
- [58] Andrey Bogdanov, Miroslav Knežević, Gregor Leander, Deniz Toz, Kerem Varici, and Ingrid Verbauwhede. Spongent: A lightweight hash function. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 312–325. Springer, September / October 2011.
- [59] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems – CHES 2007*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, September 2007.
- [60] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, François-Xavier Standaert, John P. Steinberger, and Elmar Tischhauser. Key-alternating ciphers in a provable setting: Encryption using a small number of public permutations - (extended abstract). In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 45–62. Springer, April 2012.
- [61] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knežević, Lars R. Knudsen, Gregor Leander, Ventsislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçın. PRINCE - A low-latency block cipher for pervasive computing applications -

## BIBLIOGRAPHY

---

- extended abstract. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 208–225. Springer, December 2012.
- [62] Julia Borghoff, Lars R. Knudsen, and Mathias Stolpe. Bivium as a mixed-integer linear programming problem. In Matthew G. Parker, editor, *12th IMA International Conference on Cryptography and Coding*, volume 5921 of *Lecture Notes in Computer Science*, pages 133–152. Springer, December 2009.
- [63] Gilles Brassard. On computationally secure authentication tags requiring short secret shared keys. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *Advances in Cryptology*, pages 79–86. Springer US, 1983.
- [64] Julien Bringer, Hervé Chabanne, and Emmanuelle Dottax. HB<sup>++</sup>: a lightweight authentication protocol secure against some attacks. In *Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006. SecPerU 2006. Second International Workshop on*, pages 28–33. IEEE, 2006.
- [65] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218. ACM Press, May 1998.
- [66] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, May 2001.
- [67] Christophe De Cannière, Orr Dunkelman, and Miroslav Knežević. KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems – CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, September 2009.
- [68] Leonard Carlitz. The arithmetic of polynomials in a Galois field. *Proceedings of the National Academy of Sciences*, 17(2):120–122, 1931.
- [69] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing, STOC '77*, pages 106–112, New York, NY, USA, 1977. ACM.

## BIBLIOGRAPHY

---

- [70] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.
- [71] Debrup Chakraborty, Vicente Hernandez-Jimenez, and Palash Sarkar. Another look at XCB. *Cryptology ePrint Archive*, Report 2013/823, 2013. <http://eprint.iacr.org/2013/823>.
- [72] Shan Chen, Rodolphe Lampe, Jooyoung Lee, Yannick Seurin, and John P. Steinberger. Minimizing the two-round Even-Mansour cipher. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part I*, volume 8616 of *Lecture Notes in Computer Science*, pages 39–56. Springer, August 2014.
- [73] Shan Chen and John P. Steinberger. Tight security bounds for key-alternating ciphers. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 327–350. Springer, May 2014.
- [74] Carlos Cid, Loïc Ferreira, Matthew J. B. Robshaw, and Gordon Procter. Algebraic cryptanalysis and RFID authentication. In *Radio Frequency Identification: Security and Privacy Issues*, *Lecture Notes in Computer Science*. Springer, 2015. (to appear).
- [75] Carlos Cid and Gaëtan Leurent. An analysis of the XSL algorithm. In Bimal K. Roy, editor, *Advances in Cryptology – ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 333–352. Springer, December 2005.
- [76] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Algebraic aspects of the advanced encryption standard*. Springer Science & Business Media, 2006.
- [77] Carlos Cid and Ralf-Philipp Weinmann. Block ciphers: algebraic cryptanalysis and Groebner bases. In *Groebner bases, coding, and cryptography*, pages 307–327. Springer, 2009.
- [78] Benoit Cogliati and Yannick Seurin. On the provable security of the iterated Even-Mansour cipher against related-key and chosen-key attacks. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 584–613. Springer, April 2015.

## BIBLIOGRAPHY

---

- [79] CAESAR committee. CAESAR call for submissions, draft. <http://competitions.cr.ypt.to/caesar-call-1.html>, 2013.
- [80] Jean-Sébastien Coron, Yevgeniy Dodis, Cécile Malinaud, and Prashant Puniya. Merkle-Damgård revisited: How to construct a hash function. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 430–448. Springer, August 2005.
- [81] Jean-Sébastien Coron, Jacques Patarin, and Yannick Seurin. The random oracle model and the ideal cipher model are equivalent. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 1–20. Springer, August 2008.
- [82] Nicolas Courtois. Cryptanalysis of SFINKS. In Dongho Won and Seungjoo Kim, editors, *ICISC 05: 8th International Conference on Information Security and Cryptology*, volume 3935 of *Lecture Notes in Computer Science*, pages 261–269. Springer, December 2006.
- [83] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In Bart Preneel, editor, *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Springer, May 2000.
- [84] Nicolas Courtois and Willi Meier. Algebraic attacks on stream ciphers with linear feedback. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 345–359. Springer, May 2003.
- [85] Nicolas Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Springer, December 2002.
- [86] David A Cox, John Little, and Donal O’Shea. *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer Science & Business Media, 2007.

## BIBLIOGRAPHY

---

- [87] Paul Crowley. Mercy: A fast large block cipher for disk sector encryption. In Bruce Schneier, editor, *Fast Software Encryption – FSE 2000*, volume 1978 of *Lecture Notes in Computer Science*, pages 49–63. Springer, April 2001.
- [88] Joan Daemen. Limitations of the Even-Mansour construction (rump session). In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology – ASIACRYPT’91*, volume 739 of *Lecture Notes in Computer Science*, pages 495–498. Springer, November 1993.
- [89] Joan Daemen, René Govaerts, and Joos Vandewalle. Weak keys for IDEA. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO’93*, volume 773 of *Lecture Notes in Computer Science*, pages 224–231. Springer, August 1994.
- [90] Joan Daemen and Vincent Rijmen. The block cipher Rijndael. In Jean-Jacques Quisquater and Bruce Schneier, editors, *Smart Card Research and Applications*, volume 1820 of *Lecture Notes in Computer Science*, pages 277–284. Springer Berlin Heidelberg, 2000.
- [91] Joan Daemen and Vincent Rijmen. The wide trail design strategy. In Bahram Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 222–238. Springer, December 2001.
- [92] Donald Davies and David Clayden. The message authenticator algorithm (MAA) and its implementation. NPL Report DITC 109/88, Feb 1988. Available from <http://www.cix.co.uk/~klockstone/maa.pdf>.
- [93] Jean Paul Degabriele, Victoria Fehr, Marc Fischlin, Tommaso Gagliardoni, Felix Günther, Giorgia Azzurra Marson, Arno Mittelbach, and Kenneth G. Paterson. Unpicking PLAID - A cryptographic analysis of an ISO-standards-track authentication protocol. *Cryptology ePrint Archive*, Report 2014/728, 2014. <http://eprint.iacr.org/2014/728>.
- [94] Bert den Boer. A simple and key-economical unconditional authentication scheme. *Journal of Computer Security*, 2:65–72, 1993.
- [95] Whitfield Diffie and Martin E Hellman. New directions in cryptography. *Information Theory, IEEE Transactions on*, 22(6):644–654, 1976.

## BIBLIOGRAPHY

---

- [96] Itai Dinur, Orr Dunkelman, Nathan Keller, and Adi Shamir. Key recovery attacks on 3-round Even-Mansour, 8-step LED-128, and full AES2. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 337–356. Springer, December 2013.
- [97] Dang Nguyen Duc and Kwangjo Kim. Securing HB<sup>+</sup> against GRS man-in-the-middle attack. In *Institute of Electronics, Information and Communication Engineers, Symposium on Cryptography and Information Security*, 2007.
- [98] Orr Dunkelman, Nathan Keller, and Adi Shamir. Minimalism in cryptography: The Even-Mansour scheme revisited. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 336–354. Springer, April 2012.
- [99] Morris Dworkin. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC. NIST Special Publication 800-38D, <http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf>, Nov 2007.
- [100] Morris Dworkin. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST Special Publication 800-38C, [http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C\\_updated-July20\\_2007.pdf](http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C_updated-July20_2007.pdf), Jul 2007.
- [101] Morris Dworkin. Recommendation for block cipher modes of operation: The XTS-AES mode for confidentiality on storage devices. NIST Special Publication 800-38E, <http://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf>, Jan 2010.
- [102] EMVCo. *EMV Integrated Circuit Card Specifications for Payment Systems, Book 2, Security and Key Management*, June 2008. Version 4.2.
- [103] EPCglobal. Generation 2 UHF RFID. specification for RFID air interface protocol for communications at 860 mhz 960 MHz. EPC Radio Frequency Identity Protocols Version 1.2.0, GS1, Geneva, Switzerland, 2011. <http://www.gs1.org/gsmp/kc/epcglobal/uhfc1g2>.

## BIBLIOGRAPHY

---

- [104] EPCglobal. Generation 2 UHF RFID. specification for RFID air interface protocol for communications at 860 mhz 960 MHz. EPC Radio Frequency Identity Protocols Version 2.0.0, GS1, Geneva, Switzerland, 2011. <http://www.gs1.org/gsm/kc/epcglobal/uhfc1g2>.
- [105] Mark Etzel, Sarvar Patel, and Zulfikar Ramzan. SQUARE HASH: Fast message authentication via optimized universal hash functions. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO’99*, volume 1666 of *Lecture Notes in Computer Science*, pages 234–251. Springer, August 1999.
- [106] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. In Hideki Imai, Ronald L. Rivest, and Tsutomu Matsumoto, editors, *Advances in Cryptology – ASIACRYPT’91*, volume 739 of *Lecture Notes in Computer Science*, pages 210–224. Springer, November 1993.
- [107] Shimon Even and Yishay Mansour. A construction of a cipher from a single pseudorandom permutation. *Journal of Cryptology*, 10(3):151–162, 1997.
- [108] Pooya Farshim and Gordon Procter. The related-key security of iterated even-mansour ciphers. Cryptology ePrint Archive, Report 2014/953, 2014. <http://eprint.iacr.org/2014/953>.
- [109] Pooya Farshim and Gordon Procter. The related-key security of iterated Even–Mansour ciphers. In *Fast Software Encryption*, Lecture Notes in Computer Science. Springer, 2015. (to appear).
- [110] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using gröbner bases. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 44–60. Springer, August 2003.
- [111] Niels Ferguson. Authentication weaknesses in GCM. Comments submitted to NIST modes of operation process, <http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/Ferguson2.pdf>, 2005.
- [112] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein hash function family. <http://www.skein-hash.info/sites/default/files/skein1.3.pdf>, 2010.

## BIBLIOGRAPHY

---

- [113] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of RC4. In Serge Vaudenay and Amr M. Youssef, editors, *SAC 2001: 8th Annual International Workshop on Selected Areas in Cryptography*, volume 2259 of *Lecture Notes in Computer Science*, pages 1–24. Springer, August 2001.
- [114] Institute for Electrical and Electronics Engineers. British mathematicians honoured by IEEE for technology discovery that made online shopping possible. 100<sup>th</sup> Milestone Dedicated for Significant Technical Achievement in IEEE Areas of Interest, 2010.
- [115] European Union Agency for Network and Information Security (ENISA). *Algorithms, key size and parameters report*, November 2014. [http://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014/at\\_download/fullReport](http://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014/at_download/fullReport).
- [116] A. Freier, P. Karlton, and P. Kocher. The secure sockets layer (SSL) protocol version 3.0. IETF Request for Comments 6101, <http://tools.ietf.org/html/rfc6101>, 2011.
- [117] Craig Gentry and Zulfikar Ramzan. Eliminating random permutation oracles in the Even-Mansour cipher. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 32–47. Springer, December 2004.
- [118] Benoît Gérard, Vincent Grosso, María Naya-Plasencia, and François-Xavier Standaert. Block ciphers that are easier to mask: How far can we go? In Guido Bertoni and Jean-Sébastien Coron, editors, *Cryptographic Hardware and Embedded Systems – CHES 2013*, volume 8086 of *Lecture Notes in Computer Science*, pages 383–399. Springer, August 2013.
- [119] E. N. Gilbert, F. J. MacWilliams, and N. J. A. Sloane. Codes which detect deception. Technical Report 3, Bell Sys. Tech. J., Mar 1974.
- [120] Henri Gilbert, Matthew Robshaw, and Herve Sibert. Active attack against HB<sup>+</sup>: a provably secure lightweight authentication protocol. *Electronics Letters*, 41(21):1169–1170, 2005.

## BIBLIOGRAPHY

---

- [121] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. Good variants of HB+ are hard to find. In Gene Tsudik, editor, *FC 2008: 12th International Conference on Financial Cryptography and Data Security*, volume 5143 of *Lecture Notes in Computer Science*, pages 156–170. Springer, January 2008.
- [122] Henri Gilbert, Matthew J. B. Robshaw, and Yannick Seurin. HB<sup>‡</sup>: Increasing the security and efficiency of HB<sup>+</sup>. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 361–378. Springer, April 2008.
- [123] Marc Girault, Guillaume Poupard, and Jacques Stern. On the fly authentication and signature schemes based on groups of unknown order. *Journal of Cryptology*, 19(4):463–487, October 2006.
- [124] David Goldenberg, Susan Hohenberger, Moses Liskov, Elizabeth Crump Schwartz, and Hakan Seyalioglu. On tweaking Luby-Rackoff blockciphers. In Kaoru Kurosawa, editor, *Advances in Cryptology – ASIACRYPT 2007*, volume 4833 of *Lecture Notes in Computer Science*, pages 342–356. Springer, December 2007.
- [125] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO’84*, volume 196 of *Lecture Notes in Computer Science*, pages 276–288. Springer, August 1984.
- [126] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM (JACM)*, 33(4):792–807, 1986.
- [127] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC ’82, pages 365–377, New York, NY, USA, 1982. ACM.
- [128] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.
- [129] Zheng Gong, Svetla Nikova, and Yee-Wei Law. KLEIN: a new family of lightweight block ciphers. In *RFID. Security and Privacy*, pages 1–18. Springer Berlin Heidelberg, 2011.

## BIBLIOGRAPHY

---

- [130] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON family of lightweight hash functions. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 222–239. Springer, August 2011.
- [131] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED block cipher. In Bart Preneel and Tsuyoshi Takagi, editors, *Cryptographic Hardware and Embedded Systems – CHES 2011*, volume 6917 of *Lecture Notes in Computer Science*, pages 326–341. Springer, September / October 2011.
- [132] Shai Halevi and Hugo Krawczyk. MMH: Software message authentication in the Gbit/second rates. In Eli Biham, editor, *Fast Software Encryption – FSE’97*, volume 1267 of *Lecture Notes in Computer Science*, pages 172–189. Springer, January 1997.
- [133] Dae Wan Han. Gröbner basis attacks on lightweight RFID authentication protocols. *Journal of Information Processing Systems*, 7(4):691–706, 2011.
- [134] Helena Handschuh, Lars R. Knudsen, and Matthew J. B. Robshaw. Analysis of SHA-1 in encryption mode. In David Naccache, editor, *Topics in Cryptology – CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 70–83. Springer, April 2001.
- [135] Helena Handschuh and Bart Preneel. Key-recovery attacks on universal hash function based MAC algorithms. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 144–161. Springer, August 2008.
- [136] Viet Tung Hoang and Phillip Rogaway. On generalized Feistel networks. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 613–630. Springer, August 2010.
- [137] Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bon-Seok Koo, Changhoon Lee, Donghoon Chang, Jesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. HIGHT: A new block cipher suitable for low-resource device. In Louis Goubin and Mitsuru Matsui, editors, *Cryptographic Hardware and Embedded Systems – CHES 2006*, volume 4249 of *Lecture Notes in Computer Science*, pages 46–59. Springer, October 2006.

## BIBLIOGRAPHY

---

- [138] K. Igoe and J. Solinas. AES Galois Counter Mode for the secure shell transport layer protocol. IETF Request for Comments 5647, <http://tools.ietf.org/html/rfc5647>, 2009.
- [139] International Organization for Standardization (ISO). Information technology – Automatic identification and data capture techniques – Part 13: Air interface for security services – Cryptographic suite Grain-128A. ISO/IEC Standard 29167-13, Geneva, Switzerland.
- [140] International Organization for Standardization (ISO). Information technology – Automatic identification and data capture techniques – Part 15: Air interface for security services – Crypto suite XOR. ISO/IEC Standard 29167-15, Geneva, Switzerland.
- [141] International Organization for Standardization (ISO). Information technology – Automatic identification and data capture techniques – Part 17: Air interface for security services crypto suite cryptoGPS. ISO/IEC Standard 29167-17, Geneva, Switzerland.
- [142] International Organization for Standardization (ISO). Information technology – Security techniques – Message authentication codes (MACs) – Part 1: Mechanisms using a block cipher. ISO Standard 9797-1:2011, Geneva, Switzerland, 2011.
- [143] International Organization for Standardization (ISO). Information technology – Automatic identification and data capture techniques – Part 11: Crypto suite PRESENT-80 security services for air interface communications. ISO/IEC Standard 29167-11, Geneva, Switzerland, 2014.
- [144] Tsukasa Ishiguro. Modified version of “Latin dances revisited: New analytic results of Salsa20 and ChaCha”. Cryptology ePrint Archive, Report 2012/065, 2012. <http://eprint.iacr.org/2012/065>.
- [145] Tetsu Iwata. New blockcipher modes of operation with beyond the birthday bound security. In Matthew J. B. Robshaw, editor, *Fast Software Encryption – FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 310–327. Springer, March 2006.
- [146] Tetsu Iwata and Tadayoshi Kohno. New security proofs for the 3GPP confidentiality and integrity algorithms. In Bimal K. Roy and Willi Meier, editors,

## BIBLIOGRAPHY

---

- Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 427–445. Springer, February 2004.
- [147] Tetsu Iwata and Kaoru Kurosawa. OMAC: One-key CBC MAC. In Thomas Johansson, editor, *Fast Software Encryption – FSE 2003*, volume 2887 of *Lecture Notes in Computer Science*, pages 129–153. Springer, February 2003.
- [148] Tetsu Iwata, Keisuke Ohashi, and Kazuhiko Minematsu. Breaking and repairing GCM security proofs. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 31–49. Springer, August 2012.
- [149] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In Palash Sarkar and Tetsu Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014, Part II*, volume 8874 of *Lecture Notes in Computer Science*, pages 274–288. Springer, December 2014.
- [150] Jérémy Jean, Ivica Nikolic, Thomas Peyrin, Lei Wang, and Shuang Wu. Security analysis of PRINCE. In Shiho Moriai, editor, *Fast Software Encryption – FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*, pages 92–111. Springer, March 2014.
- [151] A. Joux. On the security of blockwise secure modes of operation beyond the birthday bound. *Information Theory, IEEE Transactions on*, 56(3):1239–1246, March 2010.
- [152] Antoine Joux. Authentication failures in NIST version of GCM. Comments submitted to NIST modes of operation process, [http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/800-38\\_Series-Drafts/GCM/Joux\\_comments.pdf](http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/800-38_Series-Drafts/GCM/Joux_comments.pdf), 2006.
- [153] Ari Juels. RFID security and privacy: A research survey. *Selected Areas in Communications, IEEE Journal on*, 24(2):381–394, 2006.
- [154] Ari Juels and Stephen A. Weis. Authenticating pervasive devices with human protocols. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 293–308. Springer, August 2005.

## BIBLIOGRAPHY

---

- [155] David Kahn. *The Codebreakers: The comprehensive history of secret communication from ancient times to the internet*. Simon and Schuster, 1996.
- [156] Masanobu Katagi and Shiho Moriai. Lightweight cryptography for the internet of things. *Sony Corporation*, pages 7–10, 2008. <http://www.iab.org/wp-content/IAB-uploads/2011/03/Kaftan.pdf>.
- [157] Jonathan Katz and Yehuda Lindell. *Introduction to modern cryptography*. CRC Press, 2008.
- [158] S. Kelly and S. Frankel. Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec. IETF Request for Comments 4868, <http://www.ietf.org/rfc/rfc4868>, 2007.
- [159] S. Kelly and S. Frankel. The transport layer security (TLS) protocol version 1.2. IETF Request for Comments 5246, <http://www.ietf.org/rfc/rfc5246>, 2008.
- [160] S. Kent. IP encapsulating security payload (ESP). IETF Request for Comments 4303, <http://tools.ietf.org/html/rfc4303>, 2005.
- [161] Joe Kilian and Phillip Rogaway. How to protect DES against exhaustive key search. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO’96*, volume 1109 of *Lecture Notes in Computer Science*, pages 252–267. Springer, August 1996.
- [162] Joe Kilian and Phillip Rogaway. How to protect DES against exhaustive key search (an analysis of DESX). *Journal of Cryptology*, 14(1):17–35, 2001.
- [163] Lars Ramkilde Knudsen. Cryptanalysis of LOKI 91. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology – AUSCRYPT ’92*, volume 718 of *Lecture Notes in Computer Science*, pages 196–208. Springer Berlin Heidelberg, 1993.
- [164] Neal Koblitz and Alfred Menezes. Another look at “provable security”. Cryptology ePrint Archive, Report 2004/152, 2004. <http://eprint.iacr.org/2004/152>.
- [165] Neal Koblitz and Alfred Menezes. The random oracle model: A twenty-year retrospective. Cryptology ePrint Archive, Report 2015/140, 2015. <http://eprint.iacr.org/2015/140>.

## BIBLIOGRAPHY

---

- [166] Tadayoshi Kohno, John Viega, and Doug Whiting. CWC: A high-performance conventional authenticated encryption mode. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 408–426. Springer, February 2004.
- [167] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-hashing for message authentication. IETF Request for Comments 2104, <http://www.ietf.org/rfc/rfc2104>, 1997.
- [168] Hugo Krawczyk. LFSR-based hashing and authentication. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO’94*, volume 839 of *Lecture Notes in Computer Science*, pages 129–139. Springer, August 1994.
- [169] Hugo Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is SSL?). In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 310–331. Springer, August 2001.
- [170] Ted Krovetz and Phillip Rogaway. The software performance of authenticated-encryption modes. In Antoine Joux, editor, *Fast Software Encryption – FSE 2011*, volume 6733 of *Lecture Notes in Computer Science*, pages 306–327. Springer, February 2011.
- [171] Klaus Kursawe and Christiane Peters. Structural weaknesses in the open smart grid protocol. Cryptology ePrint Archive, Report 2015/088, 2015. <http://eprint.iacr.org/2015/088>.
- [172] Xuejia Lai and James L. Massey. A proposal for a new block encryption standard. In Ivan Damgård, editor, *Advances in Cryptology – EUROCRYPT’90*, volume 473 of *Lecture Notes in Computer Science*, pages 389–404. Springer, May 1991.
- [173] Rodolphe Lampe, Jacques Patarin, and Yannick Seurin. An asymptotically tight security analysis of the iterated Even-Mansour cipher. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 278–295. Springer, December 2012.
- [174] Rodolphe Lampe and Yannick Seurin. How to construct an ideal cipher from a small set of public permutations. In Kazue Sako and Palash Sarkar, editors,

## BIBLIOGRAPHY

---

- Advances in Cryptology – ASIACRYPT 2013, Part I*, volume 8269 of *Lecture Notes in Computer Science*, pages 444–463. Springer, December 2013.
- [175] Rodolphe Lampe and Yannick Seurin. Tweakable blockciphers with asymptotically optimal security. In Shiho Moriai, editor, *Fast Software Encryption – FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*, pages 133–151. Springer, March 2014.
- [176] Will Landecker, Thomas Shrimpton, and R. Seth Terashima. Tweakable blockciphers with beyond birthday-bound security. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 14–30. Springer, August 2012.
- [177] Will Landecker, Thomas Shrimpton, and R. Seth Terashima. Tweakable blockciphers with beyond birthday-bound security. Cryptology ePrint Archive, Report 2012/450, 2012. <http://eprint.iacr.org/2012/450>.
- [178] Adam Langley. ChaCha20 and Poly1305 for TLS. <http://www.imperialviolet.org/2013/10/07/chacha20.html>, October 2013.
- [179] Adam Langley. TLS symmetric crypto. <http://www.imperialviolet.org/2014/02/27/tlssymmetriccrypto.html>, February 2014.
- [180] IEEE Computer Society LAN/MAN Standards Committee. IEEE standard for information technology – telecommunications and information exchange between systems – local and metropolitan area networks – specific requirements. part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. amendment 6: Medium access control (MAC) security enhancements, 2004.
- [181] L. Law and J. Solinas. Suite B cryptographic suites for IPsec. IETF Request for Comments 6379, <http://tools.ietf.org/html/rfc6379>, 2011.
- [182] Éric Leveil and Pierre-Alain Fouque. An improved LPN algorithm. In Roberto De Prisco and Moti Yung, editors, *SCN 06: 5th International Conference on Security in Communication Networks*, volume 4116 of *Lecture Notes in Computer Science*, pages 348–359. Springer, September 2006.
- [183] Ticyan Li and Guilin Wang. Security analysis of two ultra-lightweight RFID authentication protocols. In Hein Venter, Mariki Eloff, Les Labuschagne, Jan

## BIBLIOGRAPHY

---

- Eloff, and Rossouw von Solms, editors, *New Approaches for Security, Privacy and Trust in Complex Environments*, volume 232 of *IFIP International Federation for Information Processing*, pages 109–120. Springer US, 2007.
- [184] Tiejian Li and Robert Deng. Vulnerability analysis of EMAP – an efficient RFID mutual authentication protocol. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, pages 238–245. IEEE, 2007.
- [185] Rudolf Lidl and Harald Niederreiter. *Finite Fields*, volume 20 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2nd edition, 1997.
- [186] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer, August 2002.
- [187] Moses Liskov, Ronald L. Rivest, and David Wagner. Tweakable block ciphers. *Journal of Cryptology*, 24(3):588–613, July 2011.
- [188] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.
- [189] Stefan Lucks. Ciphers secure against related-key attacks. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 359–370. Springer, February 2004.
- [190] David J.C. MacKay and Sanjoy Mahajan. Numbers that are sums of squares in several ways. Preprint, available from <http://www.cs.toronto.edu/~mackay/sumsquares.pdf>, 2001.
- [191] Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. In *Structure in Complexity Theory Conference, 1990, Proceedings., Fifth Annual*, pages 90–, July 1990.
- [192] Keith M. Martin. *Everyday Cryptography: Fundamental Principles and Applications*. Oxford University Press, 2012.

## BIBLIOGRAPHY

---

- [193] Mitsuru Matsui. New block encryption algorithm MISTY. In Eli Biham, editor, *Fast Software Encryption – FSE’97*, volume 1267 of *Lecture Notes in Computer Science*, pages 54–68. Springer, January 1997.
- [194] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, impossibility results on reductions, and applications to the random oracle methodology. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, February 2004.
- [195] David A. McGrew and Scott R. Fluhrer. Multiple forgery attacks against message authentication codes. Comments submitted to NIST on the choice between CWC or GCM, <http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/comments/CWC-GCM/multi-forge-01.pdf>, 2005.
- [196] David A. McGrew and John Viega. The security and performance of the Galois/Counter Mode (GCM) of operation. In Anne Canteaut and Kapalee Viswanathan, editors, *Progress in Cryptology - INDOCRYPT 2004: 5th International Conference in Cryptology in India*, volume 3348 of *Lecture Notes in Computer Science*, pages 343–355. Springer, December 2004.
- [197] David A. McGrew and John Viega. The Galois/Counter Mode of operation (GCM). Submission to NIST Modes of Operation Process, <http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/gcm/gcm-revised-spec.pdf>, May 2005.
- [198] Florian Mendel, Vincent Rijmen, Deniz Toz, and Kerem Varici. Differential analysis of the LED block cipher. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology – ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 190–207. Springer, December 2012.
- [199] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of applied cryptography*. CRC press, 1996.
- [200] Kazuhiko Minematsu. Beyond-birthday-bound security based on tweakable block cipher. In Orr Dunkelman, editor, *Fast Software Encryption – FSE 2009*, volume 5665 of *Lecture Notes in Computer Science*, pages 308–326. Springer, February 2009.

## BIBLIOGRAPHY

---

- [201] Judy H. Moore and Gustavus J. Simmons. Cycle structures of the DES with weak and semi-weak keys. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO’86*, volume 263 of *Lecture Notes in Computer Science*, pages 9–32. Springer, August 1987.
- [202] Ben Morris, Phillip Rogaway, and Till Stegers. How to encipher messages on a small domain. In Shai Halevi, editor, *Advances in Cryptology – CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 286–302. Springer, August 2009.
- [203] Jorge Munilla and Alberto Peinado. HB-MP: A further step in the HB-family of lightweight authentication protocols. *Computer Networks*, 51(9):2262–2267, 2007.
- [204] Sean Murphy and Matthew J. B. Robshaw. Essential algebraic structure within the AES. In Moti Yung, editor, *Advances in Cryptology – CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Springer, August 2002.
- [205] Chanathip Namprempre, Phillip Rogaway, and Thomas Shrimpton. Reconsidering generic composition. In Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 257–274. Springer, May 2014.
- [206] National Institute of Standards and Technology (NIST). FIPS publication 74, Guidelines for implementing and using the NBS data encryption standard, 1981.
- [207] National Institute of Standards and Technology (NIST). FIPS publication 113, Announcing the standard for computer data authentication, 2001.
- [208] National Institute of Standards and Technology (NIST). FIPS publication 197, Announcing the advanced encryption standard (AES), 2001.
- [209] National Institute of Standards and Technology (NIST). FIPS publication 198-1, The keyed-hash message authentication code (HMAC), 2008.
- [210] National Institute of Standards and Technology (NIST). FIPS publication 202, SHA-3 standard: Permutation-based hash and extendable-output functions, 2014.

## BIBLIOGRAPHY

---

- [211] Y. Nir and A. Langley. ChaCha20 and Poly1305 for IETF protocols. <http://datatracker.ietf.org/doc/draft-irtf-cfrg-chacha20-poly1305/>, Jul 2014.
- [212] National Institute of Standards and Technology. Announcing request for candidate algorithm nominations for the advanced encryption standard (AES). *Federal Register*, 62(177):48051–48058, September 1997. [http://csrc.nist.gov/archive/aes/pre-round1/aes\\_9709.htm](http://csrc.nist.gov/archive/aes/pre-round1/aes_9709.htm).
- [213] National Institute of Standards and Technology. Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA3) family. *Federal Register*, 72(212):62212–62220, November 2007. [http://csrc.nist.gov/groups/ST/hash/documents/FR\\_Notice\\_Nov07.pdf](http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf).
- [214] Information & Communication Technologies Programme of the European Commission’s Seventh Framework Programme (FP7). Ecrypt call for stream cipher primitives. <http://www.ecrypt.eu.org/stream/call/>, April 2005.
- [215] Onur Özen, Kerem Varici, Cihangir Tezcan, and Çelebi Kocair. Lightweight block ciphers revisited: Cryptanalysis of reduced round PRESENT and HIGHT. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 09: 14th Australasian Conference on Information Security and Privacy*, volume 5594 of *Lecture Notes in Computer Science*, pages 90–107. Springer, July 2009.
- [216] Daniel Panario. What do random polynomials over finite fields look like? In Gary L. Mullen, Alain Poli, and Henning Stichtenoth, editors, *Finite Fields and Applications*, volume 2948 of *Lecture Notes in Computer Science*, pages 89–108. Springer Berlin Heidelberg, 2004.
- [217] Jacques Patarin. The “coefficients H” technique (invited talk). In Roberto Maria Avanzi, Liam Keliher, and Francesco Sica, editors, *SAC 2008: 15th Annual International Workshop on Selected Areas in Cryptography*, volume 5381 of *Lecture Notes in Computer Science*, pages 328–345. Springer, August 2009.
- [218] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M Estévez-Tapiador, and Arturo Ribagorda. LMAP: A real lightweight mutual authentication pro-

## BIBLIOGRAPHY

---

- tol for low-cost RFID tags. In *Workshop on RFID security*, pages 12–14, 2006.
- [219] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M Estevez-Tapiador, and Arturo Ribagorda. M2AP: A minimalist mutual-authentication protocol for low-cost RFID tags. In *Ubiquitous Intelligence and Computing*, pages 912–923. Springer, 2006.
- [220] Pedro Peris-Lopez, Julio Cesar Hernandez-Castro, Juan M. Estevez-Tapiador, and Arturo Ribagorda. EMAP: An efficient mutual-authentication protocol for low-cost RFID tags. In Robert Meersman, Zahir Tari, and Pilar Herero, editors, *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, volume 4277 of *Lecture Notes in Computer Science*, pages 352–361. Springer Berlin Heidelberg, 2006.
- [221] Bart Preneel. Davies–Meyer. In Henk C.A. van Tilborg and Sushil Jajodia, editors, *Encyclopedia of Cryptography and Security*, pages 312–313. Springer, 2011.
- [222] Bart Preneel, René Govaerts, and Joos Vandewalle. Hash functions based on block ciphers: A synthetic approach. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO’93*, volume 773 of *Lecture Notes in Computer Science*, pages 368–378. Springer, August 1994.
- [223] Gordon Procter. A note on the CLRW2 tweakable block cipher construction. Cryptology ePrint Archive, Report 2014/111, 2014. <http://eprint.iacr.org/2014/111>.
- [224] Gordon Procter. A security analysis of the composition of ChaCha20 and Poly1305. Cryptology ePrint Archive, Report 2014/613, 2014. <http://eprint.iacr.org/2014/613>.
- [225] Gordon Procter and Carlos Cid. On weak keys and forgery attacks against polynomial-based MAC schemes. Cryptology ePrint Archive, Report 2013/144, 2013. <http://eprint.iacr.org/2013/144>.
- [226] Gordon Procter and Carlos Cid. On weak keys and forgery attacks against polynomial-based MAC schemes. In Shiho Moriai, editor, *Fast Software Encryption – FSE 2013*, volume 8424 of *Lecture Notes in Computer Science*, pages 287–304. Springer, March 2014.

## BIBLIOGRAPHY

---

- [227] Gordon Procter and Carlos Cid. On weak keys and forgery attacks against polynomial-based MAC schemes. *Journal of Cryptology*, pages 1–27, 2014.
- [228] M. O. Rabin. Fingerprinting by random polynomials. Center for Research in Computing Technology, Harvard University, Technical Report TR-15-81, 1981.
- [229] K. Raeburn. Advanced encryption standard (AES) encryption for Kerberos 5. IETF Request for Comments 3962, <http://tools.ietf.org/html/rfc3962>, 2005.
- [230] E. Rescola. The transport layer security (TLS) protocol version 1.3, draft-ietf-tls-tls13-latest. Internet Draft, <http://tlsWG.github.io/tls13-spec/>, 2015.
- [231] Thomas Ristenpart, Hovav Shacham, and Thomas Shrimpton. Careful with composition: Limitations of the indistinguishability framework. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 487–506. Springer, May 2011.
- [232] Ronald L Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [233] Matthew Robshaw and Olivier Billet. *New stream cipher designs: the eSTREAM finalists*, volume 4986. Springer Science & Business Media, 2008.
- [234] Phillip Rogaway. Practice-oriented provable security and the social construction of cryptography. Unpublished essay corresponding to an invited talk at EUROCRYPT 2009, <http://web.cs.ucdavis.edu/~rogaway/papers/cc.pdf>.
- [235] Phillip Rogaway. Bucket hashing and its application to fast message authentication. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO’95*, volume 963 of *Lecture Notes in Computer Science*, pages 29–42. Springer, August 1995.
- [236] Phillip Rogaway. Bucket hashing and its application to fast message authentication. *Journal of Cryptology*, 12(2):91–115, 1999.

## BIBLIOGRAPHY

---

- [237] Phillip Rogaway. Authenticated-encryption with associated-data. In Vijayalakshmi Atluri, editor, *ACM CCS 02: 9th Conference on Computer and Communications Security*, pages 98–107. ACM Press, November 2002.
- [238] Phillip Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC. In Pil Joong Lee, editor, *Advances in Cryptology – ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 16–31. Springer, December 2004.
- [239] Phillip Rogaway. Nonce-based symmetric encryption. In Bimal K. Roy and Willi Meier, editors, *Fast Software Encryption – FSE 2004*, volume 3017 of *Lecture Notes in Computer Science*, pages 348–359. Springer, February 2004.
- [240] Phillip Rogaway, Mihir Bellare, and John Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Trans. Inf. Syst. Secur.*, 6(3):365–403, August 2003.
- [241] Phillip Rogaway, Mihir Bellare, John Black, and Ted Krovetz. OCB: A block-cipher mode of operation for efficient authenticated encryption. In *ACM CCS 01: 8th Conference on Computer and Communications Security*, pages 196–205. ACM Press, November 2001.
- [242] Phillip Rogaway and Thomas Shrimpton. A provable-security treatment of the key-wrap problem. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 373–390. Springer, May / June 2006.
- [243] Markku-Juhani O. Saarinen. SGCM: The Sophie Germain counter mode. Cryptology ePrint Archive, Report 2011/326, 2011. <http://eprint.iacr.org/2011/326>.
- [244] Markku-Juhani Olavi Saarinen. Cycling attacks on GCM, GHASH and other polynomial MACs and hashes. In Anne Canteaut, editor, *Fast Software Encryption – FSE 2012*, volume 7549 of *Lecture Notes in Computer Science*, pages 216–225. Springer, March 2012.
- [245] M. Salter and R. Housley. Suite B profile for transport layer security (TLS). IETF Request for Comments 6460, <http://tools.ietf.org/html/rfc6460>, 2011.

## BIBLIOGRAPHY

---

- [246] Yu Sasaki<sup>1</sup>, Yosuke Todo, Kazumaro Aoki, Yusuke Naito, Takeshi Sugawara, Yumiko Murakami, Mitsuru Matsui, and Shoichi Hirose. Minalpher. Presented at DIAC, <http://info.isl.ntt.co.jp/crypt/minalpher/files/minalpher-diac2014.pdf>, 2014.
- [247] Arthur Scherbius. Cipherring machine, January 1928. US Patent 1,657,411.
- [248] Rich Schroepfel. Hasty pudding cipher specification. available from <http://web.archive.org/web/20070206162154/http://www.cs.arizona.edu/people/rcs/hpc/hpc-spec>, 1999.
- [249] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, Oct 1949.
- [250] Zhenqing Shi, Bin Zhang, Dengguo Feng, and Wenling Wu. Improved key recovery attacks on reduced-round Salsa20 and ChaCha. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology ICISC 2012*, volume 7839 of *Lecture Notes in Computer Science*, pages 337–351. Springer Berlin Heidelberg, 2013.
- [251] Victor Shoup. On fast and provably secure message authentication based on universal hashing. In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO’96*, volume 1109 of *Lecture Notes in Computer Science*, pages 313–328. Springer, August 1996.
- [252] Thomas Shrimpton and R. Seth Terashima. Personal communication, Jan 2014.
- [253] G. J. Simmons. A survey of information authentication. *Proceedings of the IEEE*, 76(5):603–620, May 1988.
- [254] Gustavus J. Simmons. Authentication theory/coding theory. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO’84*, volume 196 of *Lecture Notes in Computer Science*, pages 411–431. Springer, August 1984.
- [255] Gustavus J. Simmons. The practice of authentication. In Franz Pichler, editor, *Advances in Cryptology – EUROCRYPT’85*, volume 219 of *Lecture Notes in Computer Science*, pages 261–272. Springer, April 1986.

## BIBLIOGRAPHY

---

- [256] Gustavus J. Simmons. A Cartesian product construction for unconditionally secure authentication codes that permit arbitration. *Journal of Cryptology*, 2(2):77–104, 1990.
- [257] W. A. Stein et al. *SageMathCloud, collaborative computational mathematics*, 2015. <http://cloud.sagemath.com>.
- [258] John Steinberger. Improved security bounds for key-alternating ciphers via hellinger distance. Cryptology ePrint Archive, Report 2012/481, 2012. <http://eprint.iacr.org/2012/481>.
- [259] D. R. Stinson. The combinatorics of authentication and secrecy codes. *Journal of Cryptology*, 2(1):23–49, 1990.
- [260] D. R. Stinson. Combinatorial characterizations of authentication codes. In Joan Feigenbaum, editor, *Advances in Cryptology CRYPTO 91*, volume 576 of *Lecture Notes in Computer Science*, pages 62–73. Springer Berlin Heidelberg, 1992.
- [261] D. R. Stinson. Combinatorial characterizations of authentication codes. *Designs, Codes and Cryptography*, 2(2):175–187, 1992.
- [262] D. R. Stinson. On the connections between universal hashing, combinatorial designs and error-correcting codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(52), 1995.
- [263] Douglas R. Stinson. Universal hashing and authentication codes. *Designs, Codes and Cryptography*, 4(3):369–380, 1994.
- [264] Richard Taylor. Near optimal unconditionally secure authentication. In Alfredo De Santis, editor, *Advances in Cryptology – EUROCRYPT’94*, volume 950 of *Lecture Notes in Computer Science*, pages 244–253. Springer, May 1995.
- [265] TrueCrypt Foundation. Truecrypt user’s guide. <https://www.grc.com/misc/truecrypt/TrueCrypt%20User%20Guide.pdf>, 2012.
- [266] Serge Vaudenay. On the weak keys of Blowfish. In Dieter Gollmann, editor, *Fast Software Encryption – FSE’96*, volume 1039 of *Lecture Notes in Computer Science*, pages 27–32. Springer, February 1996.

## BIBLIOGRAPHY

---

- [267] Serge Vaudenay. Security flaws induced by CBC padding - applications to SSL, IPSEC, WTLS ... In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 534–546. Springer, April / May 2002.
- [268] G. S. Vernam. Secret signaling system, July 1919. US Patent 1,310,719.
- [269] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, 2nd edition, 2003.
- [270] Mark N. Wegman and J. Lawrence Carter. New classes and applications of hash functions. In *20th Annual Symposium on Foundations of Computer Science*, pages 175–182, 1979.
- [271] Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 22(3):265–279, 1981.
- [272] D. Whiting, R. Housley, and N. Ferguson. Counter with CBC-MAC (CCM). IETF Request for Comments 3610, <http://www.ietf.org/rfc/rfc3610>, 2003.
- [273] T. Ylonen and C. Lonvick. The secure shell (SSH) transport layer protocol. IETF Request for Comments 42563, <http://www.ietf.org/rfc/rfc4253>, 2006.
- [274] Bo Zhu, Yin Tan, and Guang Gong. Revisiting MAC forgeries, weak keys and provable security of Galois/Counter Mode of operation. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *CANS 13: 12th International Conference on Cryptology and Network Security*, volume 8257 of *Lecture Notes in Computer Science*, pages 20–38. Springer, November 2013.