Anne Canteaut
Kapaleeswaran Viswanathan (Eds.)

LNCS 3348

Progress in Cryptology – INDOCRYPT 2004

5th International Conference on Cryptology in India
Chennai, India, December 2004
Proceedings

Springer

Lecture Notes in Computer Science 3348

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Anne Canteaut
Kapaleeswaran Viswanathan (Eds.)

Progress in Cryptology – INDOCRYPT 2004

5th International Conference on Cryptology in India
Chennai, India, December 20-22, 2004
Proceedings

Springer

Volume Editors

Anne Canteaut
Institut National de Recherche en Informatique et Automatique (INRIA)
Projet CODES, Domaine de Voluceau, Rocquencourt
78153 Le Chesnay Cedex, France
E-mail: anne.canteaut@inria.fr

Kapaleeswaran Viswanathan
SETS, 21 Mangadu Swamy Street, Nungambakkam
Chennai 600 034, India
E-mail: kapali@sets.org.in

Library of Congress Control Number: 2004116723

CR Subject Classification (1998): E.3, G.2.1, D.4.6, K.6.5, F.2.1, C.2

ISSN 0302-9743
ISBN 3-540-24130-2 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11369165 06/3142 5 4 3 2 1 0

Preface

The INDOCRYPT series of conferences started in 2000. INDOCRYPT 2004 was the fifth one in this series. The popularity of this series is increasing every year. The number of papers submitted to INDOCRYPT 2004 was 181, out of which 147 papers conformed to the specifications in the call for papers and, therefore, were accepted to the review process. Those 147 submissions were spread over 22 countries.

Only 30 papers were accepted to this proceedings. We should note that many of the papers that were not accepted were of good quality but only the top 30 papers were accepted. Each submission received at least three independent reviews. The selection process also included a Web-based discussion phase. We made efforts to compare the submissions with other ongoing conferences around the world in order to ensure detection of double-submissions, which were not allowed by the call for papers. We wish to acknowledge the use of the Web-based review software developed by Bart Preneel, Wim Moreau, and Joris Claessens in conducting the review process electronically. The software greatly facilitated the Program Committee in completing the review process on time. We would like to thank Cédric Lauradoux and the team at INRIA for their total support in configuring and managing the Web-based submission and review softwares. We are unable to imagine the outcome of the review process without their participation.

This year the invited talks were presented by Prof. Colin Boyd and Prof. Amit Sahai. Colin provided a talk on the design of key establishment protocols while Amit presented a talk on secure protocols for complex tasks in complex environments. They presented two sides of the same coin so that the audience can gain a more comprehensive view of the analysis and design of cryptographic protocols. We hope that the invited talks contributed their share to promoting such an exciting area in cryptology research in India. At the same time, the invited talks were of great value for international researchers, as well, because Colin and Amit shared the latest results of their research activities.

The smooth and successful progress of INDOCRYPT 2004 was due to the efforts of many individuals. The members of the Program Committee worked hard throughout, and did an excellent job. Many external reviewers contributed their time and expertise to aid our decision-making. The Organizing Committee put its maximal effort into ensuring the successful progress of this conference. We wish to thank Prof. R. Balasubramaniam and Dr. M.S. Vijayaraghavan for being the general co-chairs of this conference. We also thank the Cryptology Research Society of India and ISI, Calcutta.

We hope that the INDOCRYPT series of conferences remains a forum for discussing high-quality results in the area of cryptology and its applications to information security in the years to come.

December 2004

Anne Canteaut
Kapaleeswaran Viswanathan

Organization

The INDOCRYPT Conferences are the annual events of the Cryptology Research Society of India. INDOCRYPT 2004 was organized by IMSc, Chennai, and SETS, Chennai.

General Co-chairs

R. Balasubramanian	Institute for Mathematical Sciences, India
M.S. Vijayaraghavan	SETS, India

Program Co-chairs

Anne Canteaut	INRIA, France
Kapaleeswaran Viswanathan	SETS, India

Program Committee

Michael Backes	IBM, Zurich, Switzerland
Colin Boyd	Queensland University of Technology, Australia
Anne Canteaut	INRIA, France
Cunsheng Ding	Hong Kong University of Science and Technology, China
Andreas Enge	Ecole Polytechnique, France
Caroline Fontaine	CNRS, France
Henri Gilbert	France Telecom R&D, France
Juanma Gonzalez-Nieto	Queensland University of Technology, Australia
Tor Helleseth	University of Bergen, Norway
Thomas Johansson	Lund University, Sweden
Kwangjo Kim	Information and Communications University, Korea
Tanja Lange	University of Bochum, Germany
Arjen Lenstra	Lucent Technologies, USA and Technische Universiteit Eindhoven, The Netherlands
C.E. Veni Madhavan	Indian Institute of Science, Bangalore, India
Keith Martin	Royal Holloway University of London, UK
Anish Mathuria	Dhirubhai Ambani, Institute of Information and Communication Technology, India

Alfred Menezes	University of Waterloo, Canada
Shiho Moriai	Sony Computer Entertainment Inc., Japan
Kenneth Paterson	Royal Holloway University of London, UK
Kapil H. Paranjape	IMSc, Chennai, India
Bart Preneel	Katholieke Universiteit Leuven, Belgium
Bimal Roy	ISI Kolkata, India
Amit Sahai	Princeton University, USA
Palash Sarkar	ISI Kolkata, India
Henk van Tilborg	Technische Universiteit Eindhoven, The Netherlands
D.G. Thomas	Madras Christian College, India
Kapaleeswaran Viswanathan	SETS, Chennai, India
Adam Young	Cigital Labs, USA
Moti Yung	Columbia University, USA

Organizing Committee

Dr. A.K. Chakravarthy	Dept. of IT, MICT, Govt. of India
Mr. Cédric Lauradoux	INRIA, France
Dr. K. Srinivas	IMSc, India
Dr. N.Vijayarangan	SETS, India

Organizing Sub-committee

Mr. G. Aswin	SETS, India
Mr. C. Stephen Balasundaram	SETS, India
Mr. Manish Chauhan	SETS, India
Ms. R. Indra	IMSc, India
Ms. K. Jayasri	SETS, India
Mr. R. Harish Kumar	SETS, India
Mr. Ramakrishna Manja	IMSc, India
Dr. Paul Pandian	IMSc, India
Mr. Vishnu Prasath	IMSc, India
Ms. A. Suganya	SETS, India
Mr. R. Vijayasarathy	SETS, India

External Referees

P.J. Abisha	Sattam Al-Riyami	Florent Bersani
Avishek Adhikari	Lejla Batina	Alex Biryukov
Riza Aditya	Côme Berbain	Simon Blackburn
Toru Akishita	Thierry Berger	Emmanuel Bresson

Jan Camenisch
Liqun Chen
Olivier Chevassut
Matthijs Coster
Deepak Kumar Dalai
V. Rajkumar Dare
Christophe De Cannière
Alex Dent
Jeroen Doumen
Dang Nguyen Duc
Sylvain Duquesne
Håkan Englund
Steven Galbraith
Pierrick Gaudry
Daniel Gottesman
Robert Granger
Kishan Chand Gupta
Darrel Hankerson
Guillaume Hanrot
Martin Hell
Clemens Heuberger
Shoichi Hirose
Yvonne Hitchcock
Dennis Hofheinz
Tetsu Iwata
Cees Jansen
Ellen Jochemsz
Stefan Katzenbeisser
Alexander Kholosha
Caroline Kudla
Joseph Lano
Hyunrok Lee
Kerstin Lemke
Benoît Libert
Vo Duc Liem
Phil MacKenzie
John Malone-Lee
Alexander Maximov
Nele Mentens
Chris Mitchell
Suman K. Mitra
François Morain
Sumio Morioka
Joern Mueller-Quade
James Muir
Svetla Nikova
Luke O'Connor
Siddika Berna Ors
Daniel Page
Matthew Parker
Olivier Pereira
Håvard Raddum
Zulfikar Ramzan
K. Rangarajan
François Recher
Akashi Satoh
Werner Schindler
Takeshi Shimoyama
Taizo Shirai
Jamshid Shokrollahi
Hervé Sibert
Francesco Sica
Andrey Sidorenko
Martijn Stam
Tsuyoshi Takagi
Gerard Tel
Yuuki Tokunaga
Ludo Tolhuizen
Emmanuel Thomé
Pim Tuyls
M.K. Viswanath
Brent Waters
Benne de Weger
Annegret Weng
Arne Winterhof
Christopher Wolf
Robbie Ye
Feng Zhu

Sponsoring Institutions

Bharat Electronics Limited
BRNS Secretariat, Bhabha Atomic Research Centre
Department of Information Technology, Government of India
Department of Science and Technology, Government of India
Electronics Corporation of India Limited
Hewlett Packard India Private Limited
Institute for Development and Research in Banking Technology
NASSCOM
SLN Technologies Private Limited
Sun Microsystems India Private Limited

Table of Contents

Invited Talks

Design of Secure Key Establishment Protocols: Successes, Failures and Prospects
Colin Boyd 1

Secure Protocols for Complex Tasks in Complex Environments
Amit Sahai 14

Cryptographic Protocols

Tripartite Key Exchange in the Canetti-Krawczyk Proof Model
Yvonne Hitchcock, Colin Boyd, Juan Manuel González Nieto 17

The Marriage Proposals Problem: Fair and Efficient Solution for Two-Party Computations
Audrey Montreuil, Jacques Patarin 33

Applications

On the Security of a Certified E-Mail Scheme
Guilin Wang, Feng Bao, Jianying Zhou 48

Multiplicative Homomorphic E-Voting
Kun Peng, Riza Aditya, Colin Boyd, Ed Dawson, Byoungcheon Lee 61

Stream Ciphers

Chosen Ciphertext Attack on a New Class of Self-Synchronizing Stream Ciphers
Bin Zhang, Hongjun Wu, Dengguo Feng, Feng Bao 73

Algebraic Attacks Over $GF(q)$
Lynn Margaret Batten 84

Cryptographic Boolean Functions

Results on Algebraic Immunity for Cryptographically Significant Boolean Functions
Deepak Kumar Dalai, Kishan Chand Gupta, Subhamoy Maitra 92

Generalized Boolean Bent Functions
Laurent Poinsot, Sami Harari 107

On Boolean Functions with Generalized Cryptographic Properties
An Braeken, Ventzislav Nikov, Svetla Nikova, Bart Preneel 120

Foundations

Information Theory and the Security of Binary Data Perturbation
Poorvi L. Vora 136

Symmetric Authentication Codes with Secrecy and Unconditionally Secure Authenticated Encryption
Luke McAven, Reihaneh Safavi–Naini, Moti Yung 148

Block Ciphers

Faster Variants of the MESH Block Ciphers
Jorge Nakahara Júnior 162

Related-Key Attacks on Reduced Rounds of SHACAL-2
Jongsung Kim, Guil Kim, Sangjin Lee, Jongin Lim, Junghwan Song 175

Related-Key Attacks on DDP Based Ciphers: CIKS-128 and CIKS-128H
Youngdai Ko, Changhoon Lee, Seokhie Hong, Jaechul Sung, Sangjin Lee 191

Cryptanalysis of Ake98
Jorge Nakahara Júnior, Daniel Santana de Freitas 206

Public Key Encryption

Designing an Efficient and Secure Public-Key Cryptosystem Based on Reducible Rank Codes
Thierry Berger, Pierre Loidreau 218

HEAD: Hybrid Encryption with Delegated Decryption Capability
Palash Sarkar 230

A Provably Secure Elliptic Curve Scheme with Fast Encryption
David Galindo, Sebastià Martín, Tsuyoshi Takagi, Jorge L. Villar 245

Efficient Representations

Advances in Alternative Non-adjacent Form Representations
Gildas Avoine, Jean Monnerat, Thomas Peyrin 260

Public Key Cryptanalysis

Attacks on Public Key Cryptosystems Based on Free Partially Commutative Monoids and Groups
Françoise Levy-dit-Vehel, Ludovic Perret 275

Exact Analysis of Montgomery Multiplication
Hisayoshi Sato, Daniel Schepers, Tsuyoshi Takagi 290

Cryptography, Connections, Cocycles and Crystals: A p-Adic Exploration of the Discrete Logarithm Problem
H. Gopalkrishna Gadiyar, KM Sangeeta Maini, R. Padma 305

Modes of Operation

EME*: Extending EME to Handle Arbitrary-Length Messages with Associated Data
Shai Halevi 315

Impossibility of Construction of OWHF and UOWHF from PGV Model Based on Block Cipher Secure Against ACPCA
Donghoon Chang, Wonil Lee, Seokhie Hong, Jaechul Sung, Sangjin Lee, Soohak Sung 328

The Security and Performance of the Galois/Counter Mode (GCM) of Operation
David A. McGrew, John Viega 343

Signatures

Revisiting Fully Distributed Proxy Signature Schemes
Javier Herranz, Germán Sáez 356

New ID-Based Threshold Signature Scheme from Bilinear Pairings
Xiaofeng Chen, Fangguo Zhang, Divyan M. Konidala, Kwangjo Kim 371

Separable Linkable Threshold Ring Signatures
Patrick P. Tsang, Victor K. Wei, Tony K. Chan, Man Ho Au, Joseph K. Liu, Duncan S. Wong 384

Traitor Tracing and Visual Cryptography

A New Black and White Visual Cryptographic Scheme for General Access Structures
Avishek Adhikari, Tridib Kumar Dutta, Bimal Roy 399

Identification Algorithms for Sequential Traitor Tracing
Marcel Fernandez, Miguel Soriano 414

Author Index 431

Design of Secure Key Establishment Protocols: Successes, Failures and Prospects

Colin Boyd*

Information Security Research Centre,
Queensland University of Technology,
Brisbane Q4001, Australia
boyd@isrc.qut.edu.au

Abstract. Key establishment protocols form one of the most basic types of cryptographic protocols and have been studied intensively for over 20 years. The current status of design and analysis methods is reviewed with particular reference to formal appoaches. Likely future trends and open issues are also discussed.

1 Introduction

Key establishment is a foundational element for secure communications. It concerns how to set up a new key (a *session key*) to protect communications during a subsequent session. In terms of modern cryptography it is a venerable problem that has been widely studied from almost every conceivable angle. One may ask how hard it can be to consider all ways of setting up a session key. Yet the evidence is that this study has not yet been exhaustive. One reason for this is that new requirements have become evident over time that were not previously recognised. Another reason is that there is no well-defined method to explore the space of possible secure protocols. Even until today most systematic or formal techniques allow only protocol analysis and not design of protocols to meet specific requirements. The purposes of this paper are:

- to explore current techniques to ensure the security of key establishment protocols, particularly those with some formal basis;
- to consider to what extent these methods can be used to systematically design new protocols;
- to summarise (and speculate on) prospects for the future of these methods.

In the rest of this introduction some background information is provided on protocol types and potential security requirements. Section 2 looks at informal design principles for key establishment. Sections 3 and 4 are devoted to the two main formal approaches to protocol analysis: the formal methods approach which

* Research funded by Australian Research Council under Discovery Project DP0345775.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 1–13, 2004.

comes from the computer security research community, and the computational approach which comes from the cryptography research community. Section 5 discusses current trends and prospects for combining the benefits of both these approaches.

1.1 Key Agreement and Key Transport

A common way of classifying key establishment is to consider protocols which provide either *key agreement* or *key transport*. Key agreement protocols require input to the session key from both parties in a two-party protocol, or more generally from more than one party in a multi-party protocol. In a key transport protocol one party (often a trusted third party) chooses the key and forwards it to the other parties.

It is often stated that key agreement is preferable to key transport. Reasons given are that key agreement is 'fairer' since no party is able to fix the key value. However, this property does not correspond to any standard security property and most models do not in any case take account of malicious insiders. Since any party is free to give away the session key at will, what may be the benefit of making the key some fixed value? In addition, it is often suggested that using pseudo-random input from more than one party serves to increase the randomness of the final key. This may or may not be useful depending on how the values are combined. In particular, suppose that two parties A and B provide values g^x and g^y in the classic Diffie-Hellman key agreement protocol. If the random number generator of A is very weak then it may be easy for an adversary to obtain x and hence the shared key g^{xy}, no matter how strong is the random number generator of B[1].

1.2 Adding Requirements

One reason that key establishment continues to be a challenging problem is the addition of new properties that are desired in certain situations. These include ways of strengthening the security properties such as the following.

Forward Secrecy is the property that compromise of long-term keys should not compromise session keys that were previously accepted. Forward secrecy is increasingly regarded as a very desirable property. It seems to be achievable only through the use of ephemeral public keys, such as in Diffie-Hellman key exchange. (Although it is not widely recognised, ephemeral keys from any public key encryption scheme can be used to provide forward secrecy, including RSA as noted by Wiener [Wie98].)

Resistance to Key Compromise Impersonation is a less widely discussed property that is related to forward secrecy in that it concerns what may happen after long-term keys are compromised. It demands that the adversary who has obtained the long-term key of entity A is unable to masquerade as other principals to A.

[1] This observation was made to me by Carsten Rudolph.

Anonymity of Principals was often neglected in the past, but with the prevalence of communications on public (including wireless) networks it is more widely recognised as an issue. For example, the Internet Key Exchange (IKE) protocol [HC98] explicitly addresses this requirement, although its provision is not so robust as may have been initially expected [PK00].

Resistance to Denial of Service is a pressing practical need for protocols, particularly those run on open networks. This is another property that was considered in the design of IKE, although there has been much controversy over the resulting solution [PK00].

As well as the above extra security features that can be relevant to any security architecture, some protocols have extra fundamental assumptions about the way that the network is set up and the security infrastructure in place.

Group Key Establishment protocols have become very popular in the recent literature in line with the increase in collaborative communications applications. There are many possible types of architecture. One of the most challenging is the ad-hoc network where the security infrastructure may be minimal.

Low-Power Principals are as prevalent as ever, due to the inexorable miniaturisation of devices. The most common example has been the mobile telephone, and there are many protocols designed specifically for its use. New lightweight technologies, such as RFID tags, open up new challenges.

Password-Based Protocols were first introduced around 15 years ago. These protocols assume that shared keys have only a small amount of entropy, and must therefore be robust against off-line guessing attacks in which the adversary attempts to eliminate potential passwords using public information. Recently such protocols have attracted extensive interest, and standards in both IEEE [IEE04] and ISO are in preparation.

Identity-Based Protocols have been around for about 20 years but recent techniques based on elliptic curve pairings have resulted in an explosion of interest in this area. These protocols allow users to establish keys without the use of an on-line server or a public key infrastructure. There is likely to be continuing interest in this area and to date few key establishment protocols using the new techniques come with a proof of security.

Notice that most combinations of the above requirements or scenarios are possible, although some are in conflict with others. For example, protocols providing forward secrecy are typically more computationally expensive than those that do not. Therefore protocols designed for low-power principals often sacrifice forward secrecy for benefits in efficiency.

2 Design Principles

In 1994 Abadi and Needham gathered together the experience of many years and produced a set of 11 rules of thumb to be used as principles for designers of cryptographic protocols [AN94]. The following year Anderson and Needham

[AN95] added a set of "robustness principles" aimed specifically at protocols in the public-key setting.

The Abadi-Needham principles can be viewed as common sense rules that can be applied in an informal protocol design process. Undoubtedly the informal design of simple protocols has benefited from wide knowledge of these rules. However, it is interesting to note that at least two, and arguably four, of the rules are about clearly defining various aspects of the protocol specification. In addition two of the seven principles of Anderson-Needham fall into this category. In other words these informal rules can be regarded as promoting the use of formality in protocol analysis.

One of the principles of Abadi and Needham can be roughly paraphrased as 'sign-before-encrypting'. In other words, when it is required to provide both authentication and confidentiality to some data, the plaintext should be signed and the result should then be encrypted. The idea behind this rule is intuitively clear: a signature of a ciphertext does not imply that the signer ever knew the plaintext. Indeed, there are several protocol attacks in which a signature on a ciphertext is removed by the adversary and replaced with a new signature. It is therefore somewhat surprising to find that many successful protocols, even those with proofs of security, ignore this rule. Paradoxically, much later analysis of the security of combining authenticity and encryption [ADR02] indicates that signing before encryption tends to give security properties no stronger than applying these operations the other way around.

3 Formal Specifications

Formal methods of specification and analysis, usually supported by software tools, have been used to analyse key establishment protocols for over 15 years. The typical analysis model uses a paradigm introduced by Dolev and Yao [DY83] in which cryptography is treated as a 'black-box' operation. This means that the adversary is able to encrypt and decrypt with any keys that it knows, but without the necessary keys will be unable to do anything with a ciphertext. Numerous formalisms and tools have been used over the years. Generally the tools search the available state space and try to establish whether insecure states can be reached. Various methods have been used to enhance the searching process. Meadows [Mea03] provides a detailed introduction to the history and progress of this research area.

3.1 Successes

There are some well-documented cases of new and unexpected attacks on protocols that have been found by machine analysis. The most celebrated is Lowe's discovery [Low96] of a flaw in the public-key protocol of Needham and Schroeder [NS78] which was found in 1996, close to 20 years after the protocol's first publication. The attack is surprisingly simple and once seen looks very obvious and not at all something beyond the capacity of a systematic search by hand.

In addition to finding flaws many protocols have been certified as free from flaws using analysis of formal specifications. Model checkers can be used to check protocols quickly and in an automated fashion. As one recent example, Basin *et al.* [BMV03] report that their 'on-the-fly model checker' (OFMC) was able to check all 36 protocols from the well known Clark–Jacob library [CJ97] in less than one minute of processing time.

3.2 Failures

A major limitation of models based on Dolev-Yao is that there is no succinct representation of the security property attained by a protocol that passes the analysis. What we know is that there is no adversary that can gain the stated secrets using the operations in the way specified. But that does not mean that there are not other strategies for the adversary that may be successful. Backes and Schunter [BS04] describe an example in which a mobile agent security protocol was formally verified to be secure with an automated theorem prover and yet it turned out to be vulnerable to a simple attack. Backes and Schunter point out that the reason for this failure was the omission of a critical action which the adversary should be allowed. Once the attack is discovered it is easy to include this action into the adversary's repertoire. A possible conclusion is that you need to already know about the potential types of attack in order to find them using the this type of model. It is perhaps harsh to regard this example as a criticism of formal methods, since protocols of the type used in this case have not yet been modelled at all using the computational models described below.

A second, and more obvious, limitation of the Dolev-Yao approach is that the cryptographic properties are not modelled faithfully. One aspect of this is that partial information leakage and probabilistic behaviour is typically ignored. A related, practically significant, issue is that different definitions of confidentiality are not distinguished. In the cryptographic community there are several different standard definitions of confidentiality including indistinguishability and non-malleability, and protection against either known plaintext or chosen plaintext attacks. Generally algorithms with stronger properties are less efficient and require stronger assumptions, so it is a good principle to use the weakest assumptions possible regarding the cryptographic algorithm required. Having found the attack on Needham-Schroeder protocol mentioned in Section 3.1, Lowe proposed an improvement which showed no weaknesses using his technique. However, neither in the original definition, nor in his improved protocol, is there a specification of the encryption algorithm to be used in terms of the standard definitions. It is not hard to see that some form of non-malleability must be provided and Lowe does point out that the adversary must not be able to alter an encrypted message. Recently Warinschi [War03] has given a computational proof assuming that the encryption algorithm has a strong security property.

3.3 Prospects

There is no doubt that research using formal methods for protocol analysis is as active as it ever has been. The plethora of tools and formalisms that were ap-

plied during the 1990s revealed new insights but it is now widely recognised that advances are required to 'go beyond Dolev-Yao' by incorporating new properties and exploring new requirements. Meadows [Mea03] provides a comprehensive review of future trends. Some of the main directions that she mentions are coverage of denial of service, anonymity, and more cryptographic properties. Meadows remarks on the trend to analyse real-world protocols, particularly those in standards. Backes and Schunter [BS04] provide a "cryptographers' wish-list" of Dolev-Yao extensions which overlaps with the issues identified by Meadows.

None of the current tools can really be used as design methods except in the sense that there are some (relatively) automatic and quick analysis tools that can be used to provide quick feedback on prototype designs. Meadows [Mea03] remarks that a possible direction towards using animation to help designers does not seem to be developing. There has been some work using tools to search for good protocols in the set of all possible protocols [CJ02]. So far it is not demonstrated that these can find useful new protocols with specified properties.

4 Provable Security

The cryptographic research community has evolved in the past 10-15 years to embrace formal foundations based on computational definitions and reductionist proofs. Acceptance of the approach is now widespread although there remain controversies [KM04], particularly when the so-called *random oracle model* is adopted. Initially the computational definitions concentrated on basic algorithms such as encryption and signature schemes. Key establishment was first considered in 1993 and interest has blossomed since the late 1990s.

4.1 Bellare–Rogaway Model

Bellare and Rogaway [BR93] initiated the computational study of key establishment in 1993. Their first paper covered only a two-party protocol between two users who already share a long-term key. Two years later [BR95] this was extended to a three-party protocol including a trusted server in the style of Needham and Schroeder's shared key protocol.

In models of this type the adversary runs the protocol in the sense that it controls which parties send and receive messages. To do this the adversary issues a Send query. The adversary has the ability to fabricate any messages that it can compute and use these as messages. In addition the adversary can obtain any session key that has been accepted by issuing a Reveal query regarding any party instance. The adversary can also issue a Corrupt query regarding any party and obtain and modify its long terms keys. These capabilities model the ability of a protocol adversary to mount replay attacks and insider attacks. The adversary eventually issues a Test query for a session that has not been opened by a Reveal or Corrupt query. The adversary's goal is to reliably distinguish between the key accepted in the test session and a random key. This is a strong definition of security but one which corresponds to the prevailing definition of security for confidentiality in encryption algorithms. The adversary is restricted only in

that it has bounded computational power; specifically it must be a probabilistic polynomial time algorithm.

Successes. By now there have been quite a few protocols proven secure in the Bellare–Rogaway model, or close variants. These include public key transport protocols, key agreement protocols, password-based protocols, multi-party key agreement and identity-based protocols. One may argue that the number of proven secure protocols is nevertheless quite small in comparison with the range of key establishment protocols currently known. Proving a protocol in this model is no small undertaking and most of the relevant papers contain proofs for only one or two protocols and require several pages of human-generated mathematical reasoning.

Failures. One criticism of the provable security approach in general is inaccessibility of the proofs. This leads in turn to a lack of wide scrutiny of the proofs [KM04]. There have been well-publicised failures in computational proofs for encryption. Proofs have also been claimed for key establishment protocols that were subsequently shown to be insecure. A protocol designed for low-power devices by Jakobsson and Pointcheval was initially published in a pre-proceedings version which was shown by Wong and Chan to be vulnerable to a simple masquerading attack [WC01]. Subsequently the protocol was fixed with a small change.

Another issue is whether protocols proven secure can be implemented in a way that they can be practically used. An important part of the security definition requires the identification of the partner of any principal in a protocol run. This is because the adversary must be forbidden from obtaining a session key in a trivial way by revealing the key from a partner who has accepted. In different versions of the Bellare–Rogaway model partnering has been defined in different ways. The most recent versions [BPR00] used the natural idea of *session identifiers*. This way of defining partners is not only intuitively clear (thus making the proofs more transparent) but also gives a practical way for entities to identify which key to use (for example on a particular communications socket). It turns out that the 1995 protocol proven secure by Bellare and Rogaway [BR95] has no reasonable way to define session identifiers. This means that although the protocol is secure it does not seem very useful. Choo *et al.* [CBHM04b] showed how a simple change to the protocol allows a natural session identifier to be defined, which can also be used in the protocol proof.

Prospects. Over the ten years and more since Bellare and Rogaway introduced their model there have been significant extensions. This has usually taken the form of new capabilities made available to the adversary to fit new requirements. For example, password-based protocols are accommodated by restricting the adversary's ability to use Send queries since each such query may be used to test a single password. Instead a new Execute query allows the adversary to observe protocol runs without trying a password guess.

Although the basic model is now firmly established, it seems likely that new variations will continue to evolve to cater for new requirements. Very recently Abdalla *et al.* [AFP04] proposed a variation in the adversary capability which allows multiple Test queries which consistently respond with the real key or a random one. Looking back at some of the additional requirements mentioned in Section 1.2 we can see that there is potential for some new additions to the model. Forward secrecy is already catered for through use of the Corrupt query, but key compromise impersonation and anonymity do not yet seem to have been modelled. These two seem to be quite achievable in this type of model, but denial of service is an area that seems to fall outside the scope of the usual computational models.

It is clear that analysis in the Bellare–Rogaway model does not provide an efficient way to design a new protocol. Varying an existing protocol is very likely to break an existing proof and there seems no useful way to guess whether a proof is possible for a new protocol.

4.2 Modular Proofs

In 1998, Bellare, Canetti and Krawczyk [BCK98] suggested a method for modular proofs of key establishment protocols. The basic idea is to first prove the protocol secure in an ideal world where messages are automatically authenticated. This ideal world is called the *authenticated links model* or simply the AM. This roughly corresponds to the situation where the adversary is passive, so unable to alter or fabricate messages (although the adversary is able to effectively delete messages). Having proved the protocol secure in the ideal world it can then be transformed into a protocol in a more realistic model in which the adversary does have the ability to fabricate messages — indeed the capabilities of the adversary are basically the same as those in the Bellare–Rogaway model.

The initial model of Bellare *et al.* [BCK98] used a security definition based on emulation between protocols in the two worlds. Later it was found that this definition is too strict to be useful and so, in 2001, Canetti and Krawczyk published a revised model [CK01] with a definition of security based on indistinguishability, similar to that of Bellare and Rogaway. Another significant benefit in the new model is that it is proven that the agreed session key can be used safely to provide secure channels, a property absent from the Bellare–Rogaway model.

Successes. The modular approach uses two types of components: the simplified protocols in the ideal world (called AM protocols) and the compilers to transform protocols into the real world (called authenticators). One of the significant benefits of the modular approach is the ability to *reuse* any AM protocol with any authenticator. Consequently, when one new component is proven secure, a whole set of new protocols results whose members are all automatically proven secure. As the number of components increases the multiplying effect of adding other components becomes more significant.

The separation of concerns between session key confidentiality and authentication also allows a much easier way to select components suitable for different applications. In other words, we may regard the modular approach as a step

towards a design method for provably secure key establishment protocols. The initial papers of Bellare *et al.* provided only a couple of examples of authenticators and AM protocols. Subsequently a number of additional examples have been provided including: a password-based authenticator [HTN+03] an authenticator based on static secrets (including identity-based secrets) [BMP04]; an AM protocol using ElGamal-type encryption [TBN03]; and an AM protocol mixing symmetric and asymmetric encryption [TVBN04].

Failures. A significant limitation of the modular approach is that it may not be possible to reach all desirable protocols by decomposing into AM protocols and authenticators. In particular, despite the existence of password-based authenticator when the server has a public key [HTN+03], password-based protocols that do not use server-public keys do not seem to allow any useful separate authenticator. This is not a limitation of the computational approach in general, since such protocols have been proven secure in the Bellare–Rogaway model [Mac02, EBP04].

A second limitation concerns the 'post-processing' of proven-secure protocols. In order to derive efficient protocols using the modular approach, it is necessary to perform some optimisation steps, particularly in the case where the AM protocol has more the one message. Currently this process is informal, so any resulting protocol strictly no longer has a security proof.

Prospects. It seems likely that more protocol components can be added to the library of existing proven secure components. This will lead to a significant number of additional protocols due to the multiplying effect mentioned above. Another likely development is the formalisation of the optimisation steps in order to make the whole process of obtaining an efficient protocol fully formal.

One direction that has not been explored yet in the Canetti and Krawczyk model is multi-party protocols (with the exception of the three-party case [HBN]). However, Katz and Yung [KY03] have proven secure a protocol compiler which works in a very similar way to an authenticator. Their compiler takes a protocol secure against a passive Bellare–Rogaway adversary (one which does not use Send queries) into one which is secure against an active adversary. It would be useful to understand the precise relationship between these related models.

5 Joining Forces

The formal methods approach to protocol analysis and the computational approach are both strong and active. In a sense their strengths and weaknesses are complementary. The formal methods approach uses an incomplete model of cryptography and lacks a transparent definition of security, but tool support gives strong assurance of analysis correctness and allows quick results to be obtained. The computational approach uses the normal definitions of cryptography, but the analysis results are slow to obtain and inaccessible to non-experts.

It is a natural goal to develop a complementary approach incorporating the strengths of both the approaches. One simple way to do this is to perform the

Dolev-Yao style of analysis using the same adversary definition as used in the computational approach, but limited to deterministic actions. Surprisingly this has only recently been explored [CBHM04a]. This process allows a hand-proven computational proof to stand alongside an automatic Dolev-Yao style analysis with a simplified model of cryptography. Going beyond this, it would be helpful to specify and explore with tools the proof process used in the computational approach. This need not take the form of a complete automatic proof checker; even a modest analysis of a part of the proof could be very beneficial. Reductionist proofs typically work by plugging a problem instance into an adversary assumed to have an advantage in breaking the protocol of interest. This requires a *simulation* of the protocol in order to let the adversary operate normally. Correct specification of the simulation seems to be an area vulnerable to errors. Therefore a formal specification and exploration of this part of the proof could be a useful way to find errors in proofs.

An alternative direction is to provide cryptographically faithful abstractions of cryptography and use these to replace the existing black-box version of cryptography. Two large research efforts which provide the potential for this are Canetti's model for universal composability [Can01] and the reactive models of Pfitzmann *et al.* [PW01]. At present these models are still too new to have seen wide application. Recently Backes [Bac04] has illustrated the potential of this approach with a hand analysis of a well-known protocol. Another effort in this direction was initiated by Abadi and Rogaway [AR02] aimed at providing a formal notion of encryption that provides a sound replacement for the usual computational definitions. Lately this has been extended by others to incorporate active adversaries [MW04]. It will be interesting to see what new insights will be gained once software tools are incorporated into this line of work.

Acknowledgements

I am grateful to Juan González and Yvonne Hitchcock for their helpful comments and suggestions.

References

[ADR02] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In *Advances in Cryptology - EUROCRYPT 2002*, pages 83–107. Springer-Verlag, 2002. Full version at `http://theory.lcs.mit.edu/~yevgen/ps/signcrypt.ps`.

[AFP04] Michel Abdalla, Pierre-Alain Fouque, and David Pointcheval. Password-based authenticated key exchange in the three-party setting. Cryptology ePrint Archive, Report 2004/233, 2004. `http://eprint.iacr.org/`.

[AN94] Martín Abadi and Roger Needham. Prudent engineering practice for cryptographic protocols. In *IEEE Symposium on Research in Security and Privacy*, pages 122–136. IEEE Computer Society Press, 1994.

[AN95] Ross Anderson and Roger Needham. Robustness principles for public key protocols. In D. Coppersmith, editor, *Advances in Cryptology – Crypto '95*, pages 236–247. Springer-Verlag, 1995. Lecture Notes in Computer Science Volume 963.

[AR02] Martín Abadi and Phillip Rogaway. Reconciling two views of cryptography (the computational soundness of formal encryption). *Journal of Cryptology*, 15(2):103–127, 2002.

[Bac04] Michael Backes. A cryptographically sounds Dolev–Yao style security proof on the Otway–Rees protocol. In *ESORICS 2004*, pages 89–108. Springer-Verlag, 2004.

[BCK98] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *30th ACM Symposium on Theory of Computing*, pages 419–428. ACM Press, 1998. Full version at `http://www-cse.ucsd.edu/users/mihir/papers/key-distribution.html`.

[BMP04] Colin Boyd, Wenbo Mao, and Kenny Paterson. Key agreement using statically keyed authenticators. In *Applied Cryptography and Network Security: Second International Conference*, pages 248–262. Springer-Verlag, 2004. Corrected version at `http://sky.fit.qut.edu.au/~boydc/papers/acns04-corrected.pdf`.

[BMV03] D. Basin, S. Mödersheim, and L. Viganò. An on-the-fly model-checker for security protocol analysis. In Einar Snekkenes and Dieter Gollmann, editors, *Proceedings of ESORICS'03*, LNCS 2808, pages 253–270. Springer-Verlag, 2003.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In B. Preneel, editor, *Advances in Cryptology – Eurocrypt 2000*, pages 139–155. Springer-Verlag, 2000. Lecture Notes in Computer Science Volume 1807.

[BR93] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In D. R. Stinson, editor, *Advances in Cryptology – Crypto '93*, pages 232–249. Springer-Verlag, 1993. Lecture Notes in Computer Science Volume 773. Full version at `http://www-cse.ucsd.edu/users/mihir`.

[BR95] Mihir Bellare and Phillip Rogaway. Provably secure session key distribution – the three party case. In *27th ACM Symposium on Theory of Computing*, pages 57–66. ACM Press, 1995.

[BS04] Michael Backes and Matthias Schunter. From absence of certain vulnerabilities towards security proofs. In *New Security Paradigms Workshop*, pages 67–74. ACM Press, 2004.

[Can01] Ran Canetti. Universally composable security: a new paradigm for cryptographic protocols (extended abstract). In IEEE, editor, *42nd IEEE Symposium on Foundations of Computer Science*, pages 136–145. IEEE Computer Society Press, 2001. Full version available at: `http://eprint.iacr.org/2000/067`.

[CBHM04a] Raymond Choo, Colin Boyd, Yvonne Hitchcock, and Greg Maitland. Complementing computational protocol analysis with formal specifications. In *Formal Aspects in Security and Trust*, Toulouse, 2004. To appear.

[CBHM04b] Raymond Choo, Colin Boyd, Yvonne Hitchcock, and Greg Maitland. On session identifiers in provably secure protocols: The bellare-rogaway three-party key distribution protocol revisited. In *Fourth Conference on Security in Communication Networks*. Springer-Verlag, 2004. To appear.

[CJ97] John Clark and Jeremy Jacob. A survey of authentication protocol literature: Version 1.0. http://www-users.cs.york.ac.uk/~jac/papers/drareview.ps.gz, November 1997.

[CJ02] John Clark and Jeremy Jacob. Protocols are programs too: the meta-heuristic search for security protocols. *Information and Software Technology*, 43:891–904, 2002.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *Advances in Cryptology – Eurocrypt 2001*, pages 453–474. Springer-Verlag, 2001. Lecture Notes in Computer Science Volume 2045. http://eprint.iacr.org/2001/040/.

[DY83] D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.

[EBP04] Olivier Chevassut Emmanuel Bresson and David Pointcheval. New security results on encrypted key exchange. In *International Workshop on Practice and Theory in Public Key Cryptography (PKC 2004)*, pages 145–158. Springer-Verlag, 2004. Also available at http://www.di.ens.fr/~pointche/pub.php?reference=BrChPo04.

[HBN] Yvonne Hitchcock, Colin Boyd, and Juan Manuel González Nieto. A password-based authenticator: Security proof and applications. In these proceedings.

[HC98] D. Harkins and D. Carrel. The internet key exchange (IKE). In *RFC 2409*. The Internet Society, 1998.

[HTN+03] Yvonne Hitchcock, Yiu Shing Terry Tin, Juan Manuel González Nieto, Colin Boyd, and Paul Montague. A password-based authenticator: Security proof and applications. In *Indocrypt 2003*, pages 388–401. Springer-Verlag, 2003. Full version at http://sky.fit.qut.edu.au/~boydc/papers/password.ps.gz.

[IEE04] IEEE. *P1363.2: Password-Based Public-Key Cryptography*, September 2004. http://grouper.ieee.org/groups/1363/passwdPK/index.html.

[KM04] Neal Koblitz and Alfred Menezes. Another look at "provable security". Cryptology ePrint Archive, Report 2004/152, 2004. http://eprint.iacr.org/.

[KY03] Jonathan Katz and Moti Yung. Scalable protocols for authenticated group key exchange. In *Advances in Cryptology – Crypto 2003*, pages 110–125. Springer-Verlag, 2003. Also available at http://www.cs.umd.edu/~jkatz/papers/multi-auth.pdf.

[Low96] Gavin Lowe. Breaking and fixing the Needham-Schroeder public key protocol using FDR. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 147–166. Springer-Verlag, 1996.

[Mac02] Philip MacKenzie. The PAK suite: Protocols for password-authenticated key exchange. Technical Report 2002-46, DIMACS, October 2002. http://dimacs.rutgers.edu/TechnicalReports/abstracts/2002/2002-46.html.

[Mea03] Catherine Meadows. Formal methods for cryptographic protocol analysis. *IEEE Journal on Selected Areas in Communications*, 21(1), 2003.

[MW04] Daniele Micciancio and Bogdan Warinschi. Soundness of formal encryption in the presence of active adversaries. In *First Theory of Cryptography Conference – TCC*, pages 133–151. Springer-Verlag, 2004.

[NS78] Roger Needham and Michael Schroeder. Using encryption for authentication in large network of computers. *Communications of the ACM*, 21:993–999, December 1978.

[PK00] Radia Perlman and Charlie Kaufman. Key exchange in IPSec: Analysis of IKE. *IEEE Internet Computing*, 4(6):50–56, November–December 2000.

[PW01] Birgit Pfitzmann and Michael Waidner. A model for asynchronous reactive systems and its application to secure message transmission. In *IEEE Symposium on Security and Privacy*, pages 184–200, 2001.

[TBN03] Yiu Shing Terry Tin, Colin Boyd, and Juan Manuel González Nieto. Provably secure mobile key exchange: Applying the Canetti-Krawczyk approach. In *Security and Privacy – ACISP 2003*, pages 166–179. Springer-Verlag, 2003.

[TVBN04] Yiu Shing Terry Tin, Harikrishna Vasanta, Colin Boyd, and Juan Manuel González Nieto. Protocols with security proofs for mobile applications. In *Security and Privacy – ACISP 2004*, pages 358–369. Springer-Verlag, 2004. Full version available at `http://sky.fit.qut.edu.au/~boydc/papers/ACISP04Full.pdf`.

[War03] B. Warinschi. A computational analysis of the needham-schroeder(-lowe) protocol. In *Proceedings of 16th Computer Science Foundation Workshop*, pages 248–262. ACM Press, 2003.

[WC01] Duncan S. Wong and Agnes H. Chan. Efficient and mutually authenticated key exchange for low power computing devices. In C. Boyd, editor, *Advances in Cryptology – Asiacrypt 2001*, pages 272–289. Springer-Verlag, 2001. Lecture Notes in Computer Science Volume 2248.

[Wie98] Michael J. Wiener. Performance comparison of public-key cryptosystems. *RSA Cryptobytes*, 4(1), 1998.

Secure Protocols for Complex Tasks in Complex Environments

Amit Sahai

University of California, Los Angeles
sahai@cs.ucla.edu

Over the last two decades, there has been tremendous success in placing cryptography on a sound theoretical foundation, and building an amazingly successful theory out of it. The key elements in this Modern Cryptographic Theory are the definitions capturing the intuitive, yet elusive notions of security in various cryptographic settings. The definitions of the early 80's proved to be extremely successful in this regard. But with time, as the theory started addressing more and more complex concerns, further notions of security had to be introduced. One of the most important concerns theory ventured into is of complex environments where different parties are communicating with each other concurrently in many different protocols. A series of efforts in extending security definitions led to the paradigm of Universally Composable (UC) Security [1], which along with modeling a general complex network of parties and providing definitions of security in that framework, provided powerful tools for building protocols satisfying such definitions.[1]

The basic underlying notion of security in the UC framework and its many predecessors is based on *simulation*. An "ideal" world is described, where all requisite tasks get accomplished securely, as if by magic. The goal of the protocol designer is to find a way to accomplish these tasks in the "real" world, so that no malicious adversary can take advantage of this substitution of ideal magic by real protocols. To formalize this, we say that for every malicious adversary $\mathcal{A}$ that tries to take advantage of the real world, there is an adversary $\mathcal{S}$ that can achieve *essentially the same results* in the ideal world. The "results" are reflected in the behavior of an *environment*. In this survey we shall refer to this notion of security as *"Environmental Security."* If a real-life protocol "Environmentally Securely realizes" a task, it ensures us that replacing the magic by reality does not open up new unforeseen threats to the system. (There may already be threats to the system even in the ideal world. But employing cryptographic primitives cannot offer a solution if the ideal system itself is badly conceived.) The ideal-world adversary $\mathcal{S}$ is called a *simulator* as it simulates the real-world behavior of $\mathcal{A}$, in the ideal world.

A major advantage of Environmentally Secure (ES) protocols, as shown in [1], is that they are "Universally Composable," i.e., roughly, if multiple copies

[1] A similar framework to UC Security was independently proposed by Pfitzmann and Waidner [5, 6]. These two frameworks are conceptually very similar, although there are a number of technical differences. We choose to concentrate on the UC framework in this survey.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 14–16, 2004.

of an ES-protocol are present in the system (in fact they could be copies of different protocols), then they collectively ES-realize the collection of the tasks they individually ES-realize. Hence we shall often refer to the framework in [1] as the ES/UC framework, or simply ES-framework or UC-framework.

Unfortunately, this notion of security turns out to be too strong to be achievable in standard settings. It has been shown that much of the interesting cryptographic tasks (including *e.g.* commitment, zero knowledge and secure multi-party computation) *cannot* be ES-realized when the adversary can control at least half the parties [1, 2, 3]. On the other hand, under a trusted set-up assumption – that there is a public reference string chosen by a completely trusted party – it is known how to build protocols for the most ambitious of cryptographic tasks (general secure multiparty computation with dishonest majority) satisfying the Environmental Security definition [4]. However, if no trusted party is assumed, then we are left with the strong impossibility results mentioned above.

We recently overcame these impossibility results in [7]. In that work, we develop secure protocols in the plain model (without any trusted set-up assumptions), by modifying the notion of security, while still retaining the composability. The new direction taken by this work opens up many interesting new questions and directions for the field of cryptographic protocols.

In this survey talk, we will outline the fundamental ideas and results leading up to the recent work mentioned above, and the many open questions that remain.

Acknowledgements. The author's research in this area has been supported by generous grants from the NSF ITR and Cybertrust programs, as well as an Alfred P. Sloan Foundation Research Fellowship.

References

1. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Electronic Colloquium on Computational Complexity (ECCC) (016): (2001) (Preliminary version in *IEEE Symposium on Foundations of Computer Science*, pages 136–145, 2001.)
2. Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.
3. R. Canetti, E. Kushilevitz, and Y. Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *EUROCRYPT*, pages 68–86, 2003.
4. R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *ACM Symposium on Theory of Computing*, pages 494–503, 2002.
5. B. Pfitzmann and M. Waidner. Composition and integrity preservation of secure reactive systems. In ACM Conference on Computer and Communications Security (CCS 2000), pp. 245–254, 2000.

6. B. Pfitzmann and M. Waidner. A Model for Asynchronous Reactive Systems and its Application to Secure Message Transmission. In IEEE Symposium on Security and Privacy, 2001.
7. Manoj Prabhakaran and Amit Sahai. New Notions of Security: Achieving Universal Composability without Trusted Setup. In *ACM Symposium on Theory of Computing*, 2004. Full version to appear in *SIAM Journal of Computing*, Special Issue for STOC 2004. Preliminary full version available at the Cryptology ePrint Archive `http://eprint.iacr.org/`.

Tripartite Key Exchange in the Canetti-Krawczyk Proof Model*

Yvonne Hitchcock, Colin Boyd, and Juan Manuel González Nieto

Information Security Research Centre, Queensland University of Technology,
GPO Box 2434, Brisbane Q 4001, Australia
{y.hitchcock, c.boyd, j.gonzaleznieto}@qut.edu.au

Abstract. A definition of secure multi-party key exchange in the Canetti-Krawczyk proof model is proposed, followed by a proof of the security of the Joux tripartite key agreement protocol according to that definition. The Joux protocol is then combined with two authentication mechanisms to produce a variety of provably secure key agreement protocols. The properties and efficiency of the Joux based protocols thus derived are then compared with each other and other published tripartite key agreement protocols. It is concluded that the Joux protocol can be used to generate efficient yet provably secure protocols.

1 Introduction

A major goal of modern cryptography is to enable two or more users on an insecure (adversary controlled) network to communicate in a confidential manner and/or ensure that such communications are authentic. In order to realize this goal, symmetric key cryptographic tools are often used due to their efficiency compared to public key techniques. However, use of such tools requires the creation of a secret key (which is typically at least 100 bits long) known only to the users communicating with each other. Because of the impracticality of each possible pair of users sharing a long term secret key, public key and/or password-based techniques are used to generate such a key when it is required. An advantage of this method of key generation is to keep different sessions independent, which enables the avoiding of replay attacks (since the wrong key will have been used for the replay) and lessens the impact of key compromise (since only one session will be exposed, not all previous communications).

Although recent progress has been made on the use of formal proof models to prove the security of key exchange protocols, one area where further work is required is the use of formal proof models in conjunction with tripartite key agreement protocols. Tripartite key agreement enables three parties to exchange a key so that they can all participate in a session. It can also be used to enable two parties to communicate in the presence of a third party who may provide

* Full version of this paper is available at http://sky.fit.qut.edu.au/∼boydc/papers/.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 17–32, 2004.

chairing, auditing, data recovery or escrow services [1]. In 2000, Joux [13] proposed a tripartite key exchange protocol based on pairings on an elliptic curve (such as the Weil or Tate pairing) that required only one round, but was subject to a man-in-the-middle attack due to its lack of any authentication mechanism. Al-Riyami and Paterson [1] have modified the Joux protocol in a variety of ways to overcome this problem, yet without adding to the number of rounds required by the protocol. However, only one of their protocols was accompanied by any sort of formal security proof, and that proof did not allow adaptive adversaries. In fact, flaws were found in preliminary versions of their protocols, demonstrating the difficulty of ensuring protocols are not flawed without the use of formal security proofs.

In this paper, we provide security proofs for Joux-based tripartite key agreement protocols that provide (implicit) key authentication. To do this, we adopt the Canetti-Krawczyk proof model [9] (hereafter referred to as the CK-model), which was based on the model of Bellare, Canetti and Krawczyk [3]. The CK-model offers the advantage of allowing modular proofs, thus allowing different components to be proven secure separately, and then joined together to produce a secure key exchange protocol. It also leads to simpler, less error-prone proofs and the ability to construct a large number of secure protocols from a much smaller number of basic secure components.

The modularity of the CK-model is gained by applying a protocol translation tool, called an *authenticator*, to protocols proven secure in a much simplified adversarial setting where authentication of the communication links is not required. The result of such an application is secure protocols in the unsimplified adversarial setting where the full capabilities of the adversary are modelled.

Unfortunately, the definition of secure key exchange provided by the original CK-model only caters for two parties, and so a modification of the definition is required to cater for tripartite key exchange. Such a modification is proposed in this paper, in conjunction with the analysis of the security and efficiency of the Joux [13] protocol. It transpires that the Joux based protocols proposed and proven secure in this paper require the same number of messages as an ordinary discrete logarithm based tripartite Diffie-Hellman protocol. However, the Joux based protocols have smaller messages and require a comparable amount of computation.

2 Overview of the Canetti-Krawczyk Approach

Here a description of the CK-model is given. Further details can be found in [3] and [9]. The CK-model defines protocol principals who may simultaneously run multiple local copies of a message driven protocol. Each local copy is called a session and has its own local state. Two sessions are *matching* if each session has the same session identifier and the purpose of each session is to establish a key between the particular two parties running the sessions. A session is *expired* if the session key agreed by the session has been erased from the session owner's memory. A powerful adversary attempts to break the protocol by in-

teracting with the principals. In addition to controlling all communications between principals, the adversary is able to *corrupt* any principal, thereby learning all information in the memory of that principal (e.g. long-term keys, session states and session keys). The adversary may impersonate a corrupted principal, although the corrupted principal itself is not activated again and produces no further output or messages. The adversary may also *reveal* internal session states or agreed session keys. The adversary must be efficient in the sense of being a probabilistic polynomial time algorithm. An *unexposed* session is one such that neither it nor a matching session has had its internal state or agreed session key revealed, and if the owner of the session or a matching session is corrupted, the corruption occurred after the key had expired at the corrupted party.

Definition 1 (Informal). *An* AKE *protocol is called* session key (SK-) secure *if the following two conditions are met. Firstly, if two uncorrupted parties complete matching sessions, then they both accept the same key. Secondly, suppose the adversary chooses as a "test session" one that is completed, unexpired and unexposed. Then if the adversary is given either the session key (in this case let* $b = 0$*) or a random string (in this case let* $b = 1$*), each with probability* $1/2$*, the probability of the adversary correctly guessing which one it received (i.e. correctly guessing the value of* b*) is not greater than 1/2 plus a negligible function in the security parameter.*

Two adversarial models are defined: the unauthenticated-links adversarial model (UM) and the authenticated-links adversarial model (AM). The UM corresponds to the "real world" where the adversary completely controls the network in use, and may modify or create messages from any party to any other party. The AM is a restricted version of the UM where the adversary may choose whether or not to deliver a message, but if a message is delivered, it must have been created by the specified sender and be delivered to the specified recipient without alteration. In addition, any such message may only be delivered once. In this way, authentication mechanisms can be separated from key agreement mechanisms by proving the key agreement secure in the AM, and then applying an authentication mechanism to the key agreement messages so that the overall protocol is secure in the UM.

An *authenticator* is a protocol translator that takes an SK-secure protocol in the AM to an SK-secure protocol in the UM. Authenticators can be constructed using one or more *message transmission (*MT*-) authenticators*. An MT-authenticator is a protocol which delivers one message in the UM in an authenticated manner. To translate an SK-secure protocol in the AM to an SK-secure protocol in the UM an MT-authenticator can be applied to each message and the resultant sub-protocols combined to form one overall SK-secure protocol in the UM. However, if the SK-secure protocol in the AM consists of more than one message, the resultant protocol is usually optimized to reduce the number and size of messages, involving reorder and reuse of message components. This practice was used in the CK-model proposal [9], although without a formal security proof.

The CK-model automatically ensures that secure protocols also provide perfect forward secrecy [9] through the use of session expiration. The CK-model also ensures that secure protocols are immune to unknown key share attacks [7–pp. 139–140]. This can be shown by contradiction. Suppose that an unknown key share attack exists on an SK-secure protocol. Then the two sessions knowing the secret key are not matching, since the identities of the supposed participants are different. Hence the adversary can choose one session as the test session and reveal the key in the other session and so achieve a non-negligible advantage. However, this contradicts the assumption of SK-security. Key compromise impersonation attacks [7–p. 52] are not covered by the CK-model since parties are unable to send or receive messages after corruption (it is only possible for the adversary to send or receive on the behalf of the corrupted party).

3 Definition of Secure Tripartite Key Exchange

The definitions of key exchange protocols and SK-security in the AM and UM provided by Canetti and Krawczyk in [9] are restricted to the case of two participants. It is necessary to extend the existing definitions to cater for at least three parties for use with tripartite key exchange. Therefore, the input to a key exchange protocol running within each party with identity P_i is redefined to be $(D, sid, \texttt{role})$, where sid is the session identifier, $D = \{P_i, P_j, P_k, \ldots\}$ is the set of identities of participants in the key exchange and $P_i \in D$. We make the new requirement that one and only one publicly available function, f, be specified (for the purpose of linking party identities to session identifiers). It is then required that for all inputs of the form $(D, sid, \texttt{role})$ to key exchange protocols, $f(D, s, d) = 1$ for some d, and $f(D', s, d') = 0$ for any set $D' \neq D$ and any d' (including $d' = d$). It is also required that sid be unique to each party using it. As an example, let $D = \{P_i, P_j, P_k\}$, let H be a collision resistant hash function, and let N_i, N_j and N_k be nonces freshly generated by P_i, P_j and P_k respectively, where the nonces are of a sufficient length that the probability of any one of them previously having been generated is negligible. Defining $s = H(P_i \parallel P_j \parallel P_k \parallel N_i \parallel N_j \parallel N_k)$ (where $\parallel$ indicates concatenation) and defining $f(\{P_i, P_j, P_k\}, s, (N_i \parallel N_j \parallel N_k)) = 1$ if and only if $s = H(P_i \parallel P_j \parallel P_k \parallel N_i \parallel N_j \parallel N_k)$ satisfies all of the above requirements.

Definition 2 (Matching). *Any u sessions (where each session is run by a different party) are* matching *if each session has the same session identifier. In particular, any two sessions with the same session identifier are said to be matching.*

Definition 3 (Session Key Security). *A t-party* KE *protocol π is called* session key (SK-) secure *in the* AM *(respectively* UM*) if the following two properties hold for any adversary $\mathcal{A}$ (respectively $\mathcal{U}$) in the* AM *(respectively* UM*).*

1. *Protocol π satisfies the property that if t uncorrupted parties complete a set of t matching sessions then they all output the same key.*
2. *The probability that $\mathcal{A}$ (respectively $\mathcal{U}$) guesses correctly the bit b from the test-session (i.e. outputs $b' = b$) is no more than $1/2$ plus a negligible function in the security parameter (i.e. no better than a random guess).*

The two requirements of the definition of SK-security in the t party case directly correspond to those in the two party case. The modified requirements regarding the session identifier are the major change in the case of key exchange involving more than two parties. The identities of the participants are linked to the session identifier to make it easy to avoid scenarios where two or more sessions have identical keys but are not matching due to different beliefs by the sessions about who is participating in the protocol.

It is worth noting that in the CK-model, since uncorrupted protocol participants always follow the protocol, a protocol participant, say A, can trust another participant, say B, to pass on correct input from a third party, say C. The first party, A, does not need to receive an authenticated message from C, but only an authenticated message from B. A trusts that B received an authenticated message from C containing the information that B forwarded to A.

4 Tripartite Key Exchange Protocol in the AM

The notation used by the tripartite key exchange protocol is as follows:

A, B, C : Protocol participants exchanging a secret key.
P : Base point of the elliptic curve.
$\mathbb{G}_1$: Points on an elliptic curve with a suitable pairing.
$\mathbb{G}_2$: Group of the same size as $\mathbb{G}_1$.
n : Order of $\mathbb{G}_1$ and $\mathbb{G}_2$.
$e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$: An admissible bilinear map (properties are below).
σ_X^Y : Signature by X intended for Y. Specifying the intended recipient clarifies the purpose of each signature in protocol descriptions, although in practice the intended recipient need not be specified.
${}^X\mathcal{E}_Y$: Encryption by X intended for Y. Specifying the sender clarifies the purpose of each encryption in protocol descriptions, although in practice the sender does not necessarily need to be specified.

An admissible bilinear map must be bilinear, non-degenerate and computable [5]. That is, the map must satisfy $e([a]P, [b]Q) = e(P, Q)^{ab}$ for all $P, Q \in \mathbb{G}_1$ and all $a, b \in \mathbb{Z}$, the map must not send all pairs in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$, and there must be an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in \mathbb{G}_1$. It is possible to construct an admissible bilinear map based on either the Weil pairing or Tate pairing over an elliptic curve [12, 5].

Joux has described an unauthenticated broadcast tripartite key exchange protocol which requires only one round [13, 19]. In the protocol, each party chooses a random value $a \in_R \mathbb{Z}_n$ and broadcasts $[a]P$ to the other two parties. The

shared key is computed as $e(P,P)^{abc}$, where a, b and c are the random values chosen by the three parties. Since no authenticators are available for broadcast protocols, two unicast versions (Protocols 1 and 2) are examined in the AM. A session identifier, *sid*, has been added to all protocol messages, and Protocol 2 uses one party to act as messenger between the other two parties to reduce the total number of message flows. The value of *sid* is not specified here, but the CK-model assumes it to be known by protocol participants before the protocol begins. In practice, the session identifier may be determined during protocol execution [9, 20]. It is assumed that messages in the AM implicitly specify sender and receiver. If it is not possible to determine the identities of all protocol participants from *sid* (e.g. the case where *sid* contains a hash of the identities), it may be necessary to include the identities of the participants in the first two messages from A. However, since all parties must ensure the correctness of *sid*, such "hints" can be omitted from the formal protocol specification.

A on input (A, B, C, sid) :	C on receipt of $(sid, [a]P)$:
$A \rightarrow B : (sid, [a]P), \quad a \in_R \mathbb{Z}_n$	$C \rightarrow A : (sid, [c]P), \quad c \in_R \mathbb{Z}_n$
$A \rightarrow C : (sid, [a]P)$	$C \rightarrow B : (sid, [c]P)$
B on receipt of $(sid, [a]P)$:	Shared Key : $e(P,P)^{abc} = e([b]P, [c]P)^a$
$B \rightarrow A : (sid, [b]P), \quad b \in_R \mathbb{Z}_n$	$= e([a]P, [c]P)^b$
$B \rightarrow C : (sid, [b]P)$	$= e([a]P, [b]P)^c$

Protocol 1: Joux protocol in the AM without broadcast messages

$B \rightarrow A : (sid, [b]P) \quad$ (where $b \in_R \mathbb{Z}_n$)
$C \rightarrow A : (sid, [c]P) \quad$ (where $c \in_R \mathbb{Z}_n$)
$A \rightarrow B : (sid, [a]P, [c]P) \quad$ (where $a \in_R \mathbb{Z}_n$)
$A \rightarrow C : (sid, [a]P, [b]P)$
$Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c$

Protocol 2: Variant of Joux protocol in the AM that can be used with authenticators to create efficient UM protocols

In order to prove the security of Protocols 1 and 2 in the AM, it is necessary to assume that the Decisional Bilinear Diffie-Hellman Problem (DBDH) is hard. The assumption has been studied by Cheon and Lee [11], and can be described similarly to the Decisional Diffie-Hellman assumption [9] as follows:

Definition 4 (DBDH assumption). *Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear map that takes as input two elements of $\mathbb{G}_1$ and outputs an element of $\mathbb{G}_2$. Let n be the order of $\mathbb{G}_1$ and $\mathbb{G}_2$, and let P be an element of $\mathbb{G}_1$. Let two probability distributions of tuples of seven elements, Q_0 and Q_1, be defined as:*

$$Q_0 = \{\langle \mathbb{G}_1, \mathbb{G}_2, P, [a]P, [b]P, [c]P, e(P,P)^{abc} \rangle : a, b, c \in_R \mathbb{Z}_n\} \text{ and}$$
$$Q_1 = \{\langle \mathbb{G}_1, \mathbb{G}_2, P, [a]P, [b]P, [c]P, e(P,P)^{d} \rangle : a, b, c, d \in_R \mathbb{Z}_n\}.$$

Then the DBDH *assumption states that* Q_0 *and* Q_1 *are computationally indistinguishable.*

Theorem 1. *Given the* DBDH *assumption, Protocols 1 and 2 are both* SK*-secure in the* AM*.*

The proof of Theorem 1 is similar to that of two party Diffie-Hellman key exchange [9] and is provided in the full version of this paper. It is possible to modify the protocol so that the use of the DBDH assumption in the proof can be replaced with the use of a random oracle. This also requires the proof to use the assumption that the Bilinear Diffie-Hellman (BDH) problem is hard. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be an admissible bilinear map that takes as input two elements of $\mathbb{G}_1$ and outputs an element of $\mathbb{G}_2$. Let n be the order of $\mathbb{G}_1$ and $\mathbb{G}_2$, and let P be an element of $\mathbb{G}_1$. Then the BDH problem [5] is to find $e(P,P)^{abc}$ when given $(\mathbb{G}_1, \mathbb{G}_2, P, [a]P, [b]P, [c]P)$, where a, b and $c \in_R \mathbb{Z}_n$. If the BDH problem is hard, there is no polynomial time algorithm to solve the BDH problem with non-negligible probability.

One way to modify the protocol to use this proof method is to combine $e(P,P)^{abc}$ with some sort of hash function to produce the key (e.g. $H\left(e\left(P,P\right)^{abc}\right)$ or a keyed hash function $H_{e(P,P)^{abc}}([a]P, [b]P, [c]P)$). The logic of the proof is based on the observation that since the hash function is completely random, the adversary can only obtain information about the session key by querying the hash function oracle with the input that would have been used to generate the session key. However, if the adversary is able to produce such a value with which to query the oracle, then the adversary is also able to break the BDH problem, which was assumed to be hard. The formal proof proceeds in a similar fashion to that of the proof using the DBDH assumption.

5 Applying Authenticators to the Joux Protocol

In order to create an SK-secure protocol in the UM, it is necessary to apply one or more authenticators to the Joux protocol. Here we focus on two authenticators originally proposed by Bellare et al. [3], λ_{SIG} (requiring the use of a signature scheme secure against adaptive chosen message attacks [17]) and λ_{ENC} (requiring the use of an encryption scheme indistinguishable under chosen ciphertext attacks [4] and a secure MAC scheme). Their specifications are given by Protocols 3 and 4.

Applying λ_{SIG} to each message of the Joux protocol (Protocol 1) results in an eighteen message protocol. However, it is possible to optimize this protocol to produce a much more efficient version. This can be done by replacing the nonce denoted by r_B in λ_{SIG} with $[a]P$, $[b]P$ or $[c]P$ (depending on which party was required to generate the nonce). In addition, in most cases, the two signatures

produced by each party can be combined to a single signature containing one copy of each of the items originally contained in the two separate signatures. Finally, only the session identifier needs to be included at the beginning of each UM message to determine to which session the messages belong. (In the specification of the MT-authenticators, the messages were unique and the entire message from the AM was included at the start of each UM message for this purpose since there were no session identifiers.) The resultant protocol in the UM is shown by Protocol 5 and requires a total of five messages and four signatures.

Protocol 3: λ_{ENC}

$$
\begin{array}{ccc}
A & & B \\
 & \underrightarrow{m} & \\
 & & N_B \in_R \{0,1\}^k \\
 & \underleftarrow{m,\ {}^{B}\mathcal{E}_A(N_B)} & \\
 & \underrightarrow{m, \text{MAC}_{N_B}(m,B)} &
\end{array}
$$

Protocol 4: λ_{SIG}

$$
\begin{array}{ccc}
A & & B \\
 & \underrightarrow{m} & \\
 & & r_B \in_R \{0,1\}^k \\
 & \underleftarrow{m, r_B} & \\
 & \underrightarrow{m, \sigma_A^B(m, r_B, B)} &
\end{array}
$$

Protocols 3 and 4: Encryption and signature-based MT-authenticators, λ_{ENC} and λ_{SIG}

$$
\begin{aligned}
&A \to B : (sid, [a]P) \quad (\text{where } a \in_R \mathbb{Z}_n) \\
&B \to C : \left(sid, [a]P, [b]P, \sigma_B^A\left(A, sid, [a]P, [b]P\right)\right) \quad (\text{where } b \in_R \mathbb{Z}_n) \\
&C \to A : \left(sid, [b]P, [c]P, \sigma_C^{A,B}\left(A, B, sid, [a]P, [b]P, [c]P\right), \sigma_B^A\left(A, sid, [a]P, [b]P\right)\right) \\
&\qquad\qquad (\text{where } c \in_R \mathbb{Z}_n) \\
&A \to B : \left(sid, [c]P, \sigma_A^{B,C}\left(B, C, sid, [a]P, [b]P, [c]P\right), \sigma_C^{A,B}\left(A, B, sid, [a]P, [b]P, [c]P\right)\right) \\
&B \to C : \left(\sigma_B^C\left(C, sid, [b]P, [c]P\right), \sigma_A^{B,C}\left(B, C, sid, [a]P, [b]P, [c]P\right)\right) \\
&\qquad Key : e(P,P)^{abc} = e([b]P,[c]P)^a = e([a]P,[c]P)^b = e([a]P,[b]P)^c
\end{aligned}
$$

Protocol 5: Joux protocol authenticated with λ_{SIG}

It is possible to combine $\sigma_B^A(A, sid, [a]P, [b]P)$ and $\sigma_B^C(C, sid, [b]P, [c]P)$ from Protocol 5 into one signature at the expense of an extra message, as shown by Protocol 6. Protocol 7 is another possible UM protocol where some messages have been combined after the authenticator has been applied to create a broadcast protocol. It has five broadcasts and three signatures.

The λ_{SIG} authenticator can be applied to Protocol 2 and then optimized to produce Protocol 8 in the UM. It requires five messages but only three signatures.

A protocol resulting from applying the λ_{ENC} authenticator to the AM Joux protocol and optimizing it is described by Protocol 9. The optimized protocol requires a total of five messages, six encryptions and six MACs. Allowing messages to be broadcast does not change these requirements.

$$
\begin{aligned}
A &\to B : (sid, [a]P) \quad (\text{where } a \in_R \mathbb{Z}_n)\\
B &\to C : (sid, [a]P, [b]P) \quad (\text{where } b \in_R \mathbb{Z}_n)\\
C &\to A : \left(sid, [b]P, [c]P, \sigma_C^{A,B}(A, B, sid, [a]P, [b]P, [c]P)\right) (\text{where } c \in_R \mathbb{Z}_n)\\
A &\to B : \left(sid, [c]P, \sigma_A^{B,C}(B, C, sid, [a]P, [b]P, [c]P), \sigma_C^{A,B}(A, B, sid, [a]P, [b]P, [c]P)\right)\\
B &\to C : \left(\sigma_B^{A,C}(A, C, sid, [a]P, [b]P, [c]P), \sigma_A^{B,C}(B, C, sid, [a]P, [b]P, [c]P)\right)\\
B/C &\to A : \sigma_B^{A,C}(A, C, sid, [a]P, [b]P, [c]P)\\
&\quad Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c
\end{aligned}
$$

Protocol 6: Joux protocol authenticated with λ_{SIG} using a minimal number of signatures

$$
\begin{aligned}
A &\to B, C : (sid, [a]P) \quad (\text{where } a \in_R \mathbb{Z}_n)\\
B &\to C, A : (sid, [b]P) \quad (\text{where } b \in_R \mathbb{Z}_n)\\
C &\to A, B : \left(sid, [c]P, \sigma_C^{A,B}(sid, A, B, [a]P, [b]P, [c]P)\right) \quad (\text{where } c \in_R \mathbb{Z}_n)\\
A &\to B, C : \left(sid, \sigma_A^{B,C}(B, C, sid, [a]P, [b]P, [c]P)\right)\\
B &\to A, C : \left(\sigma_B^{C,A}(A, C, sid, [a]P, [b]P, [c]P)\right)\\
&\quad Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c
\end{aligned}
$$

Protocol 7: Joux protocol authenticated with λ_{SIG}, broadcast version

$$
\begin{aligned}
A &\to B : (sid, [a]P) \quad (\text{where } a \in_R \mathbb{Z}_n)\\
B &\to C : \left(sid, [a]P, [b]P, \sigma_B^A(A, sid, [a]P, [b]P)\right) \quad (\text{where } b \in_R \mathbb{Z}_n)\\
C &\to A : \left(sid, [b]P, [c]P, \sigma_C^A(A, sid, [a]P, [c]P), \sigma_B^A(A, sid, [a]P, [b]P)\right)\\
&\qquad\qquad (\text{where } c \in_R \mathbb{Z}_n)\\
A &\to B : \left(sid, [c]P, \sigma_A^{B,C}(B, C, sid, [a]P, [b]P, [c]P)\right)\\
A \text{ or } B &\to C : \left(sid, [c]P, \sigma_A^{B,C}(B, C, sid, [a]P, [b]P, [c]P)\right)\\
&\quad Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c
\end{aligned}
$$

Protocol 8: Variant of Joux protocol authenticated with λ_{SIG}

$$
\begin{aligned}
A &\to B : sid, [a]P, \ {}^A\mathcal{E}_B(N_{AB}), \ {}^A\mathcal{E}_C(N_{AC})\\
B &\to C : sid, [a]P, [b]P, \ {}^B\mathcal{E}_C(N_{BC}), \ {}^B\mathcal{E}_A(N_{BA}), \mathrm{MAC}_{N_{AB}}(sid, [b]P, A), \ {}^A\mathcal{E}_C(N_{AC})\\
C &\to A : sid, [b]P, [c]P, \ {}^C\mathcal{E}_A(N_{CA}), \ {}^C\mathcal{E}_B(N_{CB}), \mathrm{MAC}_{N_{AC}}(sid, [c]P, A),\\
&\qquad \mathrm{MAC}_{N_{BC}}(sid, [c]P, B), \ {}^B\mathcal{E}_A(N_{BA}), \mathrm{MAC}_{N_{AB}}(sid, [b]P, A)\\
A &\to B : sid, [c]P, \mathrm{MAC}_{N_{BA}}(sid, [a]P, B), \mathrm{MAC}_{N_{CA}}(sid, [a]P, C), \ {}^C\mathcal{E}_B(N_{CB}),\\
&\qquad \mathrm{MAC}_{N_{BC}}(sid, [c]P, B)\\
B &\to C : sid, \mathrm{MAC}_{N_{CB}}(sid, [b]P, C), \mathrm{MAC}_{N_{CA}}(sid, [a]P, C)\\
&\quad Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c
\end{aligned}
$$

Protocol 9: Joux protocol authenticated with λ_{ENC}

Another protocol using λ_{ENC} can be constructed in the UM, by using the variant of the Joux protocol in the AM (Protocol 2). The optimized version is shown by Protocol 10 and requires five messages, four encryptions and four MACs. In a broadcast version of the protocol, the last two messages can be combined into one broadcast so that only four messages are required. However, the same number of encryptions and MACs are still required by the broadcast version.

$$
\begin{aligned}
A &\rightarrow B : sid, [a]P,\ {}^{A}\mathcal{E}_B(N_{AB}),\ {}^{A}\mathcal{E}_C(N_{AC}) \\
B &\rightarrow C : sid, [a]P, [b]P,\ {}^{B}\mathcal{E}_A(N_{BA}),\ \text{MAC}_{N_{AB}}(sid, [b]P, A),\ {}^{A}\mathcal{E}_C(N_{AC}) \\
C &\rightarrow A : sid, [b]P, [c]P,\ {}^{C}\mathcal{E}_A(N_{CA}),\ \text{MAC}_{N_{AC}}(sid, [c]P, A), \\
&\qquad {}^{B}\mathcal{E}_A(N_{BA}),\ \text{MAC}_{N_{AB}}(sid, [b]P, A) \\
A &\rightarrow B : sid, [c]P,\ \text{MAC}_{N_{BA}}(sid, C, [a]P, [c]P, B),\ \text{MAC}_{N_{CA}}(sid, [a]P, [b]P, C) \\
A/B &\rightarrow C : sid,\ \text{MAC}_{N_{CA}}(sid, [a]P, [b]P, C) \\
&\qquad Key : e(P,P)^{abc} = e([b]P, [c]P)^a = e([a]P, [c]P)^b = e([a]P, [b]P)^c
\end{aligned}
$$

Protocol 10: Variant of Joux protocol authenticated with λ_{ENC}

6 Efficiency of Joux Based Protocols in the UM

Table 1 shows the efficiency of each of the different optimized protocols in the UM described in Section 5. The table shows that the efficiency of each scheme depends heavily on the signature or encryption scheme chosen for the implementation. Since the protocols will be executed using an elliptic curve where a pairing is available that can be used as the basis of an admissible bilinear map (as defined at the beginning of Section 4), the suitability of various pairing-based signature and encryption schemes for the above protocols has been investigated. A brief description of each scheme is included below, and the efficiency of each scheme summarized in Tables 2 and 3.

Table 1. Operations and messages required by tripartite UM protocols

Protocol number	5	6	8	8	7	9	10	10
Broadcast used	N	N	N	Y	Y	N/Y	N	Y
Messages	5	6	5	4	5	5	5	4
Signatures	4	3	3	3	3	-	-	-
Verifications	6	6	4	4	6	-	-	-
Encryptions	-	-	-	-	-	6	4	4
Decryptions	-	-	-	-	-	6	4	4
MACs	-	-	-	-	-	6	4	4
Scalar mults.	3	3	3	3	3	3	3	3
Exponentiations	3	3	3	3	3	3	3	3
Pairings	3	3	3	3	3	3	3	3

Table 2. Efficiency of signature schemes using pairings

Scheme	Signature				Verification			
	Pair.	Exp.	Sc. mul.	Other	Pair.	Exp.	Sc. mul.	Other
Hess	-	1	1		1+(1)	1	-	
BLS	-	-	1		2	-	-	
CC	-	-	2		2	-	1	
LQ	(1)	2	1.4	symm. enc.	2+(2)	1	-	symm. dec.
M-L	(1)	-	3		3+(1)	1	-	
SOK	-	-	2		2 or 3	-	-	

(y) indicates an additional y operations required in a precomputation.

M-L. This identity-based signcryption scheme was proposed by Malone-Lee [16]. A summary of its efficiency is also provided by Nalla and Reddy [18]. The scheme provides non-repudiation if the plaintext is surrendered to the party required to perform an independent verification. The scheme can therefore be used in place of a signature scheme if desired.

NR. This identity-based encryption scheme (requiring a trusted authority) was proposed by Nalla and Reddy [18]. The trusted authority can also be used to provide non-repudiation.

BLS. This scheme to provide short signatures was proposed by Boneh, Lynn and Shacham [6]. The scheme is not identity-based and allows different signatures to be combined, thus saving bandwidth (at the expense of extra computation). The scheme can alse be used for batch verification, to increase verification efficiency if several users sign the same message.

Lynn. This scheme to provide authenticated identity-based encryption was proposed by Lynn [15]. The scheme does not provide non-repudiation [14] and so can not be used in place of a signature scheme. It is noteworthy that this scheme actually uses fewer pairings than the BF scheme which provides encryption only. However, the Lynn scheme does require use of symmetric encryption and decryption algorithms.

CC. This scheme to provide an identity-based signature was proposed by Cha and Cheon [10].

Hess. This scheme was proposed by Hess [12] and is an identity-based signature scheme. The paper also includes a comparison with the CC and SOK schemes.

SOK. This scheme was proposed by Sakai, Ohgishi and Kasahara and an efficiency analysis is provided by Hess [12].

BF. This identity-based encryption scheme was proposed by Boneh and Franklin [5].

LQ. This identity-based signcryption scheme was proposed by Libert and Quisquater [14]. It can require the use of a symmetric encryption and decryption scheme, and can be used as either a signature or an encryption scheme

since it provides non-repudiation because any party can verify the origin of the ciphertext. However, verification of the origin of the plaintext requires the key used for the symmetric encryption to be provided to the party performing the verification. Another property of the scheme is that the symmetric encryption and decryption can be replaced by some extra modular multiplications if the plaintext to be encrypted is only short. The signcryption requires a total of two scalar multiplications, but these can be performed together in the time of about 1.4 scalar multiplications.

Although the signcryption schemes can be used as either a signature or encryption schemes, care must be taken when performing an efficiency analysis of the resulting UM protocol, since extra signatures may need to be created if a single signature was intended for use by more than one recipient in the original UM protocol.

Table 3. Efficiency of encryption schemes using pairings

Scheme	Encryption				Decryption			
	Pair.	Exp.	Sc. mul.	Other	Pair.	Exp.	Sc. mul.	Other
Lynn	(1)	-	-	symm. enc.	(1)	-	-	symm. dec.
BF	(1)	1	1		1	-	1	
NR	(1)	1	2		2+(1)	1	-	
				option				option
LQ	(1)	2	1.4	symm. enc.	2+(2)	1	-	symm. dec.
M-L	(1)	-	3		3+(1)	1	-	

(y) indicates an additional y operations required in a precomputation.

Since the pairing operation is the most expensive of those performed by the signature and encryption schemes under consideration, the authenticated identity-based encryption scheme of Lynn appears to be the most promising from an efficiency viewpoint. Combining it with the protocol requiring the least number of operations, Protocol 10, leads to an implementation of the Joux protocol in the UM requiring three on-line pairings to compute the key (one per party) and four off-line pairings. Four instead of eight off-line pairings are required since some of the off-line pairings can be reused and need not be calculated twice.

Table 4 provides a comparison of the number of operations required by Protocol 10 and those required by the tripartite protocols proposed by Al-Riyami and Paterson [1] and based on Joux's protocol, TAK-1 to TAK-4 and TAKC. The table shows that those protocols using broadcast messages (TAK-1 to TAK-4) only require 3 messages, which is less than the most efficient of the protocols proposed here. However, such protocols do not provide message authentication, only implicit key authentication. Protocol 10 has the advantage that parties accepting a secret key can be sure that the messages upon which they acted were not generated by a malicious party or replays of old messages; the other parties actually participated in the key exchange.

The non-broadcast protocol from [1] (TAKC) is designed to provide key confirmation as well as key authentication. This protocol requires more messages

than Protocol 10 because it provides key confirmation, which is not provided by Protocol 10. It can be seen that if the time for the precomputation of pairings required by Protocol 10 is ignored (since the precomputation is reusable for any key exchange involving the same participants), Protocol 10 is generally more efficient in terms of number of operations than those of [1], and requires fewer messages than the TAKC protocol, but more messages than the TAK-1 to TAK-4 protocols. In addition, Protocol 10 has an associated proof of security, whereas the only TAK protocol that currently has an associated proof of security is the TAK-1 protocol, but its proof is restricted because it does not allow the adversary to make any Reveal queries, and therefore does not cater for known session-key attacks.

Table 4. Operations and messages required by Al-Riyami and Paterson's tripartite protocols compared with Protocol 10

Protocol name	TAK-1	TAK-2	TAK-3	TAK-4	TAKC	10	10
Broadcast used	Y	Y	Y	Y	N	N	Y
Messages	3	3	3	3	6	5	4
Signatures	-	-	-	-	3	-	-
Verifications	-	-	-	-	6	-	-
Symmetric encryptions	-	-	-	-	3	4	4
Symmetric decryptions	-	-	-	-	6	4	4
MACs	-	-	-	-	-	4	4
Scalar mults.	[3]	[3]	[3]	6 + [3]	[3]	[3]	[3]
Exponentiations	$3 + \langle 3 \rangle$	6	$3 + \langle 3 \rangle$	3	3	3	3
Pairings	$3 + \langle 3 \rangle$	9	$6 + \langle 3 \rangle$	3	3	3 + (4)	3 + (4)

$y + \langle x \rangle$ indicates a total of $y + x$ operations are required, but x operations may be precomputed if identities and long term keys of participants known in advance. A new precomputation is required for each key exchange.
$y + (x)$ indicates a total of $y + x$ operations are required, but x operations may be precomputed if identities of participants known in advance. The precomputation is reusable for any key exchange involving those participants.
$[x]$ indicates that x operations may be precomputed, but a new precomputation is required for each key exchange.

It is also possible to compare Protocol 10 with existing schemes for group key exchange based on the ordinary use of discrete logarithms, such as that of Bresson, Chevassut, Pointcheval and Quisquater [8], herein denoted the BCPQ scheme. This scheme can be converted to a tripartite key exchange protocol requiring eight exponentiations (two of which can be precomputed), three signatures and four verifications. If a signature scheme such as DSA is used, signing takes one exponentiation (which can be precomputed) and verification takes two simultaneous exponentiations, or about the time of 1.2 single exponentiations. Thus the BCPQ scheme takes the total time of 10.8 online exponentiations and 5 offline exponentiations, whereas Protocol 10 requires 3 exponentiations

and 3 pairings online. Therefore, if a pairing can be computed in the time of 2.6 exponentiations, the Joux based scheme will be as efficient in terms of online computation as the BCPQ scheme. Figures due to Barreto, Kim, Lynn and Scott [2] indicate that a 512 bit pairing takes about 2.5 times as long as a 1024 bit exponentiation with a 1007 bit exponent (20ms for a pairing compared to 7.9ms for an RSA signature) or 4.9 times as long as a 1024 bit exponentiation with a 160 bit exponent (20ms for a pairing compared to 4.09ms for a DSA signature). Thus Protocol 10 compares favourably to the BCPQ scheme if a large exponent is used with that scheme, but not if a small exponent is used. However, there has recently been a substantial amount of research on improving pairing efficiency, and it is possible that the efficiency of pairings may improve to the extent that Protocol 10 is more efficient than the BCPQ scheme for small exponents also.

7 Conclusion

The CK-model has been used to examine the security of tripartite key exchange protocols based on the Joux protocol. A new definition of security for key exchange protocols with more than two participants has been provided, and a proof of security for the Joux protocol in the AM given. The efficiency of the UM protocols created by combining the Joux AM protocol with signature and encryption based authenticators has been analysed, and the efficiency of various pairing based encryption and signature schemes which could be used in the authentication mechanism has been summarized. It has been concluded that a secure tripartite key exchange protocol can be formed that requires three online and four off-line pairings. This protocol also compared favourably with other published tripartite key agreement protocols.

Acknowledgements

This research is part of an ARC SPIRT project (C10024103) undertaken jointly by Queensland University of Technology and Motorola.

References

1. Sattam S. Al-Riyami and Kenneth G. Paterson. Tripartite authenticated key agreement protocols from pairings. In *Cryptography and Coding*, volume 2898 of *Lecture Notes in Computer Science*, pages 332–359. Springer-Verlag, 2003.
2. Paulo S. L. M. Barreto, Hae Y. Kim, Ben Lynn, and Michael Scott. Efficient algorithms for pairing-based cryptosystems. In *Advances in Cryptology—CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 354–368. Springer-Verlag, 2002.

3. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC '98)*, pages 419–428, New York, May 1998. ACM Press. [Full paper online] `http://www-cse.ucsd.edu/users/mihir/papers/modular.ps.gz`.
4. Mihir Bellare, Anand Desai, David Pointcheval, and Phil Rogaway. Relations among notions of security for public-key encryption schemes (extended abstract). In *Advances in Cryptology—CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 26–45. Springer-Verlag, 1998. [Full paper online] `http://www-cse.ucsd.edu/users/mihir/papers/relations.pdf`.
5. Dan Boneh and Matthew Franklin. Identity-based encryption from the Weil pairing. In *Advances in Cryptology—CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001. [Full paper online] `http://crypto.stanford.edu/~dabo/abstracts/ibe.html`.
6. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology—ASIACRYPT 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, 2001. [Full paper online] `http://crypto.stanford.edu/~dabo/abstracts/weilsigs.html`.
7. Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer-Verlag, Berlin, 2003.
8. E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater. Provably authenticated group Diffie-Hellman key exchange. In P. Samarati, editor, *Proc. of ACM-CCS 01*, pages 255–264, Philadelphia, Pennsylvania, USA, November 2001. ACM, ACM Press.
9. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology—EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 451–472. Springer-Verlag, 2001. [Full paper online] `http://eprint.iacr.org/2001/040.ps.gz`.
10. Jae Choon Cha and Jung Hee Cheon. An identity-based signature from gap Diffie-Hellman groups. In *Practice and Theory in Public Key Cryptography—PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer-Verlag, 2003.
11. Jung Hee Cheon and Dong Hoon Lee. Diffie-Hellman problems and bilinear maps. Cryptology ePrint Archive, Report 2002/117, 2002. [Online] `http://eprint.iacr.org/` [accessed 11/07/2003].
12. Florian Hess. Efficient identity based signature schemes based on pairings. In *Selected Areas in Cryptography—SAC 2002*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer-Verlag, 2002.
13. Antoine Joux. A one round protocol for tripartite Diffie-Hellman. In *Algorithmic Number Theory: Fourth International Symposium—ANTS-IV 2000, Proceedings*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–393. Springer-Verlag, 2000.
14. Benoît Libert and Jean-Jacques Quisquater. New identity based signcryption schemes from pairings. Cryptology ePrint Archive, Report 2003/023, 2003. [Online] `http://eprint.iacr.org/` [accessed 11/07/2003].
15. Ben Lynn. Authenticated identity-based encryption. Cryptology ePrint Archive, Report 2002/072, 2002. [Online] `http://eprint.iacr.org/` [accessed 11/07/2003].
16. John Malone-Lee. Identity-based signcryption. Cryptology ePrint Archive, Report 2002/098, 2002. [Online] `http://eprint.iacr.org/` [accessed 11/07/2003].

17. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
18. Divya Nalla and K.C. Reddy. Signcryption scheme for identity-based cryptosystems. Cryptology ePrint Archive, Report 2003/066, 2003. [Online] `http://eprint.iacr.org/` [accessed 11/07/2003].
19. K.G. Paterson. Cryptography from pairings: a snapshot of current research. *Information Security Technical Report*, 7(3):41–54, 2002.
20. Yiu Shing Terry Tin, Colin Boyd, and Juan Manuel González Nieto. Provably secure mobile key exchange: Applying the Canetti-Krawczyk approach. In *Information Security and Privacy—ACISP 2003*, volume 2727 of *Lecture Notes in Computer Science*, pages 166–179. Springer-Verlag, 2003.

The Marriage Proposals Problem: Fair and Efficient Solution for Two-Party Computations

Audrey Montreuil and Jacques Patarin

Université de Versailles,
45 avenue des Etats-Unis,
78035 Versailles Cedex - France

Abstract. In this paper we will present a fair and efficient solution to *The Marriage Proposals Problem* (i.e. two-party computation of AND). This solution uses many similar ideas with the solution to *The Socialist Millionaires' Problem* of [6] (we deal here with AND instead of EQUALITY and this introduces some practical small changes). Then we generalize our algorithm in three directions : first, to compute the AND with many players (not only two). Second, to compute any binary operators (boolean function of two inputs). In all these solutions we do not use Mix and Match techniques [20] but direct solutions based on the Diffie-Hellman assumption (whereas the solution of *The Socialist Millionaires' Problem* of [6], as Mix and Match techniques, requires the Decision Diffie-Hellman assumption). Moreover, with our solutions we have to compute less exponentiations compared with Mix and Match techniques ($50 + 4k$ instead of $78+4k$ or $96+4k$, where k is the security parameter i.e. security is in $1/2^k$, we reduce the overall security to the Diffie-Hellman problem is difficult). Third, we will explain how to have a fair computation of any boolean function with any number of inputs (i.e. any number of players) by using Mix and Match techniques (here we will explain how to extend the scheme of [20] for fair computations).

1 Introduction

Alice and Bob never met before and wish to find out whether they have some particular mutual interest. But naturally each refuses to show interest first, because of the risk of getting an embarrassing "no" from the other. More formally, Alice has a secret bit X and Bob has a secret bit Y and a protocol is needed that reveals exactly the logical AND of the two bits. Consequently, if Bob's bit is one, he cannot fail to learn Alice's bit because in that case her bit has the same value as the AND.

We wish to find a protocol which:

- convinces the two parties of the correctness of the result.
- does not allow one party whose answer is "no" to get any additional information about the other party's answer.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 33–47, 2004.

– is fair, i.e. one party cannot get the result while preventing the other one from getting it.

Alice and Bob can take one of several approaches. They might confide their bits to a trusted third party charged with the task of determining honestly whether $AND(X, Y) = 1$. However this trusted party will know X and Y.

They might also use physical means such as a trusted common computer (they can destroy it after obtaining the result in order to be sure that their inputs are not stored) or use tamper proof devices, etc. However here the software used must be trusted and Alice and Bob must be at the same place.

Another way to achieve the desired protocol is by using cards. For instance *The Five Card Trick* [4] in which two parties may securely compute the AND function based on the ability to make an oblivious cut on a deck of cards, or *Discreet Solitary Games* [14] which provides a scenario of a single person using cryptographic techniques as building blocks for playing sophisticated solitary games with cards.

In this paper we look for pure algorithmic solutions based on cryptographic tools (Alice and Bob do not have to be at the same place and do not need a common computer, they can use their own computers and communications).

Yao [26] introduced the concept of secure computation, and demonstrated a protocol that enables Alice and Bob to compute any function (here the AND) of their respective inputs without leaking to each other any additional information. Yao's protocol is both generic and efficient. The protocol is applied to a circuit that evaluates the function, and its overhead is linear in the size of the circuit. However, this protocol does not guarantee fairness. Namely, one of the parties might learn her output before the other party does, and can then terminate the protocol prematurely, before the other party learns his output.

Constructions that achieve fairness for general two-party and multiparty protocols were introduced by Yao [26], Galil, Haber and Yung [15], Brickell, Chaum, Damgård and van de Graaf [8], Ben-Or, Goldreich, Micali and Rivest [3] and Goldwasser and Levin [19]. These constructions concentrate on generic properties rather than on efficiency. The construction of [26] does not change the computation of every gate, but rather works by

– the parties generating a trapdoor function
– the circuit computing an output encrypted by that function
– the parties gradually revealing the bit of the trapdoor function.

This solution is not very efficient due to overhead of secure distributed generation and computation of the trapdoor function. A more efficient construction was given in [6] for a specific two-party function (testing the equality of two inputs): the protocol we propose in section 2 uses many similar ideas with this solution of *The Socialist Millionaires' Problem*, due to F. Boudot, B. Schoenmakers and J. Traoré, which is a variant of the *Millionaires' problem* introduced by Yao [25, 26]. In *The Millionaires' Problem* we deal with the $\geq$ function, while in *The Socialist Millionaires' Problem* we deal with EQUALITY and for *The Marriage Proposals Problem* we deal with AND. This introduces some practical changes for the most efficient solutions.

For solving the fairness issue, in all these constructions, the output of the protocol is *commitments* to Alice's and Bob's respective outputs. The commitment that becomes to Alice is made using a key known to Bob, and vice versa. After exchanging the commitments the parties open them bit by bit, ensuring that none of them gains a significant advantage over the other party. All the constructions require such a final phase in which the outputs are revealed bit by bit. Note that unlike Pinkas' protocol [21] these constructions (and the one we present in section 2) do not use the more recent *timed commitments* of Boneh and Naor [5], or the variant suggested by Garay and Jakobsson [17], which prevent a more powerful adversary from running a parallel attack for breaking commitments (note that the first application of timed commitments to fair exchange was in [24]).

We extend our protocol to n players in section 4. We propose solutions to compute any functions with two inputs in section 5. And finally we present in section 6 a fair solution based on the Diffie-Hellman assumption to compute any boolean function using Mix and Match techniques. This extends the schemes of [20] for fair computations. Another solution based on the quadratic residuosity assumption was described in [7].

The tools needed for this paper are shortly recalled with small typeface (Schnorr's protocol, etc.), excepted the *Mix and Match* techniques which we only recall in extended version of this paper. So all the proofs are given and this paper is self-contained.

2 Protocol - Fair Computation of "AND"

2.1 Assumption

Our result requires the Diffie-Hellman assumption (which implies the Discrete Logarithm assumption).

The Discrete Logarithm (DL) assumption for group G_q states that it is infeasible to compute $log_g y$ given random g, $y \in G_q$, $g \neq 1$. Or, more formally, for all constants c and for all sufficiently large q, there exists no probabilistic polynomial time Turing machine which, on input G_q, g, y, outputs $log_g y$ with probability greater than $1/|q|^c$.

The Diffie-Hellman (DH) assumption for group G_q states that it is infeasible to compute g^{ab} given random generators g, y_1, $y_2 \in G_q$, where $a = log_g y_1$ and $b = log_g y_2$. Or, more concisely, it is infeasible to compute g^{ab} given g, g^a, g^b for random $a, b \in \mathbb{Z}_q$.

Remark. The protocol of [6] requires the Decision Diffie-Hellman assumption (which implies the Diffie-Hellman assumption) because an adversary must not be able to **partially** recover the secret value X (or Y) which is in $\mathbb{Z}_q$. We do not need the Decision Diffie-Hellman assumption in our protocol, we only require the Diffie-Hellman assumption because X and Y are in $\{0, 1\}$.

The Decision Diffie-Hellman (DDH) assumption for group G_q states that it is infeasible to decide whether $y = g^{ab}$ given random generators g, y, y_1, $y_2 \in G_q$,

where $a = log_g y_1$ and $b = log_g y_2$. Or, more concisely, it is infeasible to decide whether $c = ab$ (which is equivalent to $g^c = g^{ab}$) given g, g^a, g^b, g^c for random $a, b, c \in \mathbb{Z}_q$.

2.2 Parameter Generation

Alice and Bob jointly generate a group G_q of a large prime order q. G_q must be chosen such that Diffie-Hellman problems on G_q are assumed to be infeasible. This implies that $q \geq 160$ bits. For example G_q can be taken as a subgroup of $\mathbb{Z}_p^*$ for a prime p of 1024 bits such that $q/p-1$, or alternatively G_q can be taken as a group of order q of points on an elliptic curve. They also decide on generators g_0, g_1, g_2 of G_q for which they do not know $log_{g_i} g_j$ for $i \neq j$, $0 \leq i, j \leq 2$ (nobody knows these values). Their inputs are X and Y respectively ($X, Y \in \{0, 1\}$, they want to compute $AND(X, Y)$). Let k be a security parameter, such that it is computationally infeasible to do 2^k computations in a human-scale time and with human-scale computation resources (nowadays k is generally taken ≥ 80). We need that $2^k \leq q$.

2.3 Development of the Protocol

Step 1: Alice randomly chooses $x_a \in \mathbb{Z}_q^*$ and generates $g_a = g_1^{x_a}$ and sends it to Bob. Similarly, Bob generates $g_b = g_1^{x_b}$ for a random $x_b \in \mathbb{Z}_q^*$ and sends it to Alice. They use Schnorr's protocol [23] to prove knowledge of x_a and x_b respectively. This protocol is shortly recalled here.

Schnorr's protocol on our values:
This protocol allows Alice to prove to Bob that she knows $x_a \in \mathbb{Z}_q$ satisfying $g_a = g_1^{x_a}$. Alice randomly selects an integer $r \in \mathbb{Z}_q$, computes $W = g_1^r$, $c = h(W)$ (where $h : \{0, 1\}^* \to \mathbb{Z}_q$ denote a cryptographic hash function) and $D = r - x_a c \bmod q$. Then Alice sends (c, D) to Bob.
Bob is convinced if $c = h(g_1^D g_a^c)$.

They also check that $g_a \neq 1$ and $g_b \neq 1$. They both compute $g_3 = g_1^{x_a x_b} = g_a^{x_b} = g_b^{x_a}$.

Step 2: Alice selects a random element $a \in \mathbb{Z}_q$ and a random number e, $0 \leq e < 2^k$, and computes: $(P_a, Q_a) = (g_3^a g_0^e, g_1^a g_2^{\bar{X}})$ (1) where $\bar{X} = \neg(X) = NOT(X)$ and X is the input of Alice.

Remark: Here we use $\bar{X}$ instead of X for *The Socialist Millionaires' Problem* of [6].

She sends this couple (P_a, Q_a) to Bob and a proof that it is correctly formed (i.e. that it satisfies (1)). For this proof she uses an extension of Chaum-Pedersen protocol ([12]) to the case involving several generators and a proof that a coordinate is equal to 0 or 1 (protocol constructed using the technique due to Cramer, Damgård and Schoenmakers [13]).

Extension of Chaum-Pedersen protocol on our values:
This protocol allows Alice to prove to Bob that she knows $a, e, \bar{X}$ satisfying $P_a = g_3^a g_0^e$ and $Q_a = g_1^a g_2^{\bar{X}}$. Alice randomly selects $r_a, r_e, r_{\bar{X}} \in \mathbb{Z}_q$, computes $W_1 = g_3^{r_a} g_0^{r_e}$, $W_2 = g_1^{r_a} g_2^{r_{\bar{X}}}$, $c = h(W_1, W_2)$ (where $h : \{0,1\}^* \to \mathbb{Z}_q$ denote a cryptographic hash function), $D_a = r_a - ac \bmod q$, $D_e = r_e - ec \bmod q$ and $D_{\bar{X}} = r_{\bar{X}} - \bar{X}c \bmod q$. Then Alice sends $(c, D_a, D_e, D_{\bar{X}})$ to Bob.
Bob is convinced if $c = h(g_3^{D_a} g_0^{D_e} P_a^c, g_1^{D_a} g_2^{D_{\bar{X}}} Q_a^c)$.

Protocol constructed using the technique due to Cramer, Damgård and Schoenmakers on our values:
This protocol allows Alice to prove to Bob that she knows $a \in \mathbb{Z}_q$ and $\bar{X} \in \{0,1\}$ satisfying $Q_a = g_1^a g_2^{\bar{X}}$ (in particular no information on $\bar{X}$ other than the fact that it is in $\{0,1\}$). Alice randomly selects $r, c_{1-\bar{X}}, D_{1-\bar{X}} \in \mathbb{Z}_q$, computes $W_{\bar{X}} = g_1^r$, $W_{1-\bar{X}} = g_1^{D_{1-\bar{X}}} (Q_a / g_2^{1-\bar{X}})^{c_{1-\bar{X}}}$, $c = h(W_0, W_1)$ (where $h : \{0,1\}^* \to \mathbb{Z}_q$ denote a cryptographic hash function), $c_{\bar{X}} = c - c_{1-\bar{X}} \bmod q$ and $D_{\bar{X}} = r - ac_{\bar{X}} \bmod q$. Then Alice sends the proof (c_0, c_1, D_0, D_1) to Bob.
Bob is convinced if $c_0 + c_1 = h(g_1^{D_0} Q_a^{c_0}, g_1^{D_1} (Q_a / g_2)^{c_1}) \bmod q$.

- **Version Without Fairness.** Alice sets $e = 0$ and $P_a = g_3^a$.
- **Fair Version.** Alice chooses k random values $a_i \in \mathbb{Z}_q$ and k random bits $e_i \in \{0,1\}$, $i = 0, \ldots, k-1$, subject to the condition that

$$a = \sum_{i=0}^{k-1} a_i 2^i \bmod q \quad \text{and} \quad e = \sum_{i=0}^{k-1} e_i 2^i$$

  and sets $A_i = g_3^{a_i} g_0^{e_i}$, $i = 0, \ldots, k-1$ (2). She sends A_i, $i = 0, \ldots, k-1$ to Bob with a proof that these A_i values are correctly formed (i.e. that they satisfy (2) with $e_i \in \{0,1\}$). For this proof she uses a proof that a coordinate is equal to 0 or 1 (protocol constructed using the technique due to Cramer, Damgård and Schoenmakers [13]).

Protocol constructed using the technique due to Cramer, Damgård and Schoenmakers on our values:
This protocol allows Alice to prove to Bob that she knows $a_i \in \mathbb{Z}_q$ and $e_i \in \{0,1\}$ satisfying $A_i = g_3^{a_i} g_0^{e_i}$ (in particular no information on e_i other than the fact that it is in $\{0,1\}$). Alice randomly selects $r, c_{1-e_i}, D_{1-e_i} \in \mathbb{Z}_q$, computes $W_{e_i} = g_3^r$, $W_{1-e_i} = g_3^{D_{1-e_i}} (A_i / g_0^{1-e_i})^{c_{1-e_i}}$, $c = h(W_0, W_1)$ (where $h : \{0,1\}^* \to \mathbb{Z}_q$ denote a cryptographic hash function), $c_{e_i} = c - c_{1-e_i} \bmod q$ and $D_{e_i} = r - a_i c_{e_i} \bmod q$. Then Alice sends the proof (c_0, c_1, D_0, D_1) to Bob. Bob is convinced if $c_0 + c_1 = h(g_3^{D_0} A_i^{c_0}, g_3^{D_1} (A_i / g_0)^{c_1}) \bmod q$.

Bob verifies the proofs and checks that $P_a = \prod_{i=0}^{k-1} A_i^{2^i}$.

By symmetry, he does the same as Alice:
He computes (P_b, Q_b) satisfying $(P_b, Q_b) = (g_3^b g_0^f, g_1^b g_2^{\bar{Y}})$ where $b \in \mathbb{Z}_q$ and f, $0 \le f < 2^k$, are randomly chosen.

- **Version Without Fairness.** He sets $f = 0$ and $P_b = g_3^b$.
- **Fair Version.** He chooses random $b_i \in \mathbb{Z}_q$ and $f_i \in \{0,1\}$, $i = 0, \ldots, k-1$, subject to the condition that

$$b = \sum_{i=0}^{k-1} b_i 2^i \bmod q \quad \text{and} \quad f = \sum_{i=0}^{k-1} f_i 2^i.$$

He sends $B_i = g_3^{b_i} g_0^{f_i}$ to Alice who verifies the proofs and checks that $P_b = \prod_{i=0}^{k-1} B_i^{2^i}$.

Step 3: Alice and Bob both compute $(P_a P_b, Q_a Q_b) = (g_3^{a+b} g_0^{e+f}, g_1^{a+b} g_2^{\bar{X}+\bar{Y}})$ (1)

Remark: Here we compute $P_a P_b$ and $Q_a Q_b$ instead of P_a/P_b and Q_a/Q_b for *The Socialist Millionaires' Problem* of [6].

Then Alice produces $R_a = (Q_a Q_b)^{x_a}$ and sends R_a to Bob with a proof that $log_{g_1} g_a = log_{Q_a Q_b} R_a$ to show that R_a is correctly formed (it is the same x_a than in step 1). For this proof she uses Chaum-Pedersen protocol [12].

Chaum-Pedersen protocol on our values:
This protocol allows Alice to prove to Bob that she knows $x_a \in \mathbb{Z}_q$ satisfying $R_a = (Q_a Q_b)^{x_a}$ and $g_a = g_1^{x_a}$. Alice randomly selects an integer $r \in \mathbb{Z}_q$, computes $W_1 = (Q_a Q_b)^r$, $W_2 = g_1^r$, $c = h(W_1, W_2)$ (where $h : \{0,1\}^* \to \mathbb{Z}_q$ denote a cryptographic hash function) and $D = r - x_a c \bmod q$. Then Alice sends the proof (c, D) to Bob.

Bob is convinced if $c = h((Q_a Q_b)^D R_a^c, g_1^D g_a^c)$.

Similarly Bob produces $R_b = (Q_a Q_b)^{x_b}$ with the corresponding proof. Now Alice and Bob both know on account of (3) and the definition of g_3 that

$$R_{ab} = R_a^{x_b} = R_b^{x_a} = (Q_a Q_b)^{x_a x_b} = g_3^{a+b} g_2^{(\bar{X}+\bar{Y}) x_a x_b}. \tag{4}$$

Step 4:

- **Version Without Fairness.** Alice and Bob test whether $P_a P_b = R_{ab}$ because, from (3) and (4), this equality is true if and only if $X = Y = 1$.
- **Fair Version.** Alice and Bob fairly disclose the values of e and f bit by bit but without disclosing the values of a and b. They execute the following step for $i = k-1, \ldots, 1$. They send each other the values of a_i, e_i and b_i, f_i, respectively. Bob checks that $A_i = g_3^{a_i} g_0^{e_i}$ and Alice does a similar check for b_i, f_i. Subsequently, they respectively release only e_0 and f_0. Finally, Alice proves that she knows $log_{g_3} A_0 / g_0^{e_0}$ and Bob gives a similar proof for f_0. This convinces the other party that the bits e_0 and f_0 are correct. Once the values of e and f are released both Alice and Bob may determine whether $X = Y = 1$ by testing whether $P_a P_b = R_{ab} g_0^{e+f}$.

This step can be regarded as fair, because if Bob (for example) deliberately aborts the protocol at $i = l$ say, he will be only at most one bit ahead of Alice to test, by exhaustive search, the combinations for the missing bits $e_0, \ldots e_l$.

3 Security Analysis

3.1 Definitions

In this section, we will prove the security with respect to **correctness**, **privacy** and **fairness**, against **passive** and **active attacks**, under the Diffie-Hellman assumption. These proofs are similar with the proofs of [6]. However, there are small changes due to the fact that we study the AND instead of *The Socialist Millionaires' Problem* of [6].

Correctness: At the end of the protocol Alice and Bob are convinced of the validity of the result. We have achieved this in section 2 by the (non-interactive) proofs of knowledge given by Alice and Bob, which show that the values exchanged in the various protocol steps are of the intended form. This implies that test (of step 4) at the end of the protocol is correct.
Privacy (or Secrecy): The protocol hides the private inputs of Alice and Bob, which are X and Y respectively (here X and Y are in $\{0, 1\}$ so unlike in [6] we cannot split these values in smaller parts).
Fairness: One party cannot get the result while preventing the other one from getting it.
Passive Attacks: The (passive) adversary correctly follows the specifications of the protocol. This is a modelisation of attacks that take place after the protocol has been completed, and may involve either Alice or Bob.
Active Attacks (i.e. "Malleability-Style Attacks"): The adversary may be active during the protocol and deviate from it. In particular, he does not necessary make random choices when it is prescribed in the definition of the protocol. Security against such an adversary means that there is no strategy that increases the amount of information that this adversary learns about the secret of the other party.

We will consider Bob as the adversary and Alice as the honest party. However, we could reverse these roles as our results are symmetric in nature and hides information both ways.

3.2 Security Against Passive Attacks

We recall that, clearly, if Bob's input is $Y = 1$, he will learn Alice's secret. If $Y = 0$, we have to show that Bob learns no information about Alice's input X. As the (non-interactive) proofs (of knowledge) used during this protocol are zero-knowledge, the only information learnt by Bob are the following values produced by Alice: $g_a = g_1^{x_a}$, $P_a = g_3^a g_0^e$, $Q_a = g_1^a g_2^{\bar{X}}$, $R_a = (Q_a Q_b)^{x_a}$, e. Since e is released by Alice, and Bob knows x_b, b and Y, Bob essentially learns: $g_1^{x_a}$, $T = g_1^{a x_a} = \left(\frac{P_a}{g_0^e}\right)^{x_b^{-1}}$, $g_1^a g_2^{\bar{X}+\bar{Y}}$, $g_2^{(\bar{X}+\bar{Y})x_a} = \frac{R_a}{T}\left(\frac{P_b}{g_0^f}\right)^{x_b^{-1}}$. Writing $g_1 = g$, $g_2^{\bar{X}+\bar{Y}} = g^w$, $x_a = u$, $a = v$, and reordering, we may summarize this by saying that for a generator g, Bob essentially learns g^u, g^{uv}, g^{v+w}, g^{uw}.

To prove that the protocol is secure against passive attacks that recover the value of X, we must prove that Bob is not able to compute X from these values. For this, it is sufficient to prove that he is not able to compute $g_2^{\bar{X}+\bar{Y}} = g^w$ from g^u, g^{uv}, g^{v+w}, g^{uw}. By using the following lemma, we conclude that under the Diffie-Hellman assumption, the protocol is secure against passive attacks that recover the value of X.

Lemma 1. *Under the DH assumption it is infeasible to compute g^w from g^u, g^{uv}, g^{v+w}, g^{uw}, for random $u, v, w, \in \mathbb{Z}_q$.*

Proof. Suppose we have an oracle computing g^w given g^u, g^{uv}, g^{v+w}, g^{uw}, for random $u, v, w, \in \mathbb{Z}_q$. Then we show how to compute g^b given $\alpha = g^a$, $\beta = g^{ab}$ for random $a, b \in \mathbb{Z}_q$, hence contradicting the DH assumption. The reduction is as follows. Set $\gamma = g^{ac}$, for random $c \in \mathbb{Z}_q$, and give α, β, g^c, $\frac{\gamma}{\beta}$ to the oracle. Since this tuple is equal to g^a, g^{ab}, g^c, $g^{a(c-b)}$, the oracle returns g^{c-b}, from which we obtain $\frac{g^c}{g^{c-b}} = g^b$.

3.3 Security Against Active Attacks

Here we will extend the argument developed above to the case of active adversaries. We will show that our protocol remains secure even in this case, provided that g_2 is jointly computed by Alice and Bob. The random oracle model, formalized by Bellare and Rogaway [2], will be used to model the behavior of the hash function h underlying the various non-interactive zero-knowledge proofs of our protocol. In this model, the hash function is replaced by an oracle which produces a truly random value (in the range of the function) when queried. For identical queries, the same answers are given. Various cryptographic schemes using hash functions have been proved secure in this model. In particular, Pointcheval and Stern [22] provided security proofs for signature schemes derived from *honest-verifier zero-knowledge* identification schemes.

In our proof below, all parties (including the adversary) will be modelled by probabilistic polynomial time interactive Turing machines with access to the random oracle. We will prove the following result.

Lemma 2. *Under the DH assumption and assuming the random oracle model, the (modified) protocol for the* Marriage Proposals Problem *is secure against active attacks that recover Alice's secret.*

To prove this lemma it suffices to consider the version of the protocol that does not address fairness, as the additions to make the protocol fair are not essential to our argument below.

For our proof, we need to slightly modify the original protocol. We require that Alice and Bob, in addition to g_3, jointly compute g_2. So Alice generates $g_{a_2} = g_1^{x_{a_2}}$ for random $x_{a_2} \in \mathbb{Z}_q^*$. Similarly, Bob generates $g_{b_2} = g_1^{x_{b_2}}$ for random $x_{b_2} \in \mathbb{Z}_q^*$. They use Schnorr's protocol to prove knowledge of x_{a_2} and x_{b_2} respectively. They also check that $g_{a_2} \neq 1$ and $g_{b_2} \neq 1$. Let $g_2 = g_1^{x_{a_2} x_{b_2}} = g_{a_2}^{x_{b_2}} = g_{b_2}^{x_{a_2}}$, which can be computed by both Alice and Bob.

We are now assuming that Bob is the active adversary (the analysis of the case in which Alice is the adversary is essentially the same). This means that Bob will choose his values (x_b, x_{b_2} and b) in a 'clever' way rather than truly random as specified in the protocol description. However, the messages he will send will be in accordance with the protocol. Our security proof is based on a reduction argument; we prove that if an active adversary Bob (viewed as a probabilistic polynomial time interactive Turing machine) can find, with non-negligible probability, X, hence g_2^X, given the information 'seen' during the execution of the protocol, then this adversary can be used to build a probabilistic polynomial time interactive Turing machine which contradicts the DH assumption. Here the probability is taken over the random tapes of Alice and Bob, the random oracles, the public parameters G_q, g_1 and X. For simplicity, we will not write in the sequel the dependencies on the security parameter $|q|$, but when we say that an expression f is non-negligible, this means that f depends on $|q|$ and that there exists a positive integer c such that $f(|q|)$ is larger than $1/|q|^c$ for all sufficiently large $|q|$.

Proof. Let g_1, $\alpha = g_1^{x_a}$, $\beta = g_1^{ax_a}$ (where x_a and a are random and unknown) be an instance of the DH problem (see also section 2.1). We want to obtain $\gamma = g_1^a$. We will see how we can use Bob to compute this value. We will 'convert' this instance to an input to our protocol, and exhibit a *simulator* $\mathcal{S}$ (probabilistic polynomial time interactive Turing machine) capable of simulating the three steps of our protocol in such a way that the adversary Bob cannot distinguish a real interaction with Alice from a simulated one. Bob will be used as a *resettable black box*. In other words, the *simulator* will have control over its tapes, and will have the ability to bring Bob to a halt and restart it in its staring state at any time it wishes. All the simulations will be performed under the random oracle model. $\mathcal{S}$ will play Alice's role and will speak first in the protocol.

Step 1. $\mathcal{S}$ sends α to Bob. Since $\mathcal{S}$ does not know x_a, the proof required at this step is simulated. In the random oracle model, where $\mathcal{S}$ has a full control of the values returned by the oracle, this proof can easily be simulated. In order to produce this proof, $\mathcal{S}$ randomly chooses $c \in \mathbb{Z}_q$ and $D \in \mathbb{Z}_q$. $\mathcal{S}$ then defines the output of the random oracle on the input (query) $W = g_1^D \alpha^c$ to be c (which means that $c = h(W)$). Then $\mathcal{S}$ produces tuples (c, D) with an distribution identical to the one produced by a real prover knowing x_a. This is due to the *honest-verifier zero-knowledge* property (*special honest-verifier zero-knowledge* in fact [13]) of Schnorr's interactive protocol. Then Bob sends g_b to Alice along with a proof of knowledge (c_b, D_b) (in the random oracle model) of x_b the discrete logarithm of g_b to the base g_1; the *proof* (c_b, D_b) is correct if $c_b = h(g_1^{D_b} g_b^{c_b})$, where c_b corresponds to the answer of the random oracle to the query $g_1^{D_b} g_b^{c_b}$. If this proof is not correct $\mathcal{S}$ aborts the protocol. At this point $\mathcal{S}$ needs to obtain the discrete logarithm x_b in order to carry on with its simulation. By using the technique developed by Pointcheval and Stern [22] and known as the *oracle replay attack* (forking lemma) one can easily obtain this value: if we replay Bob, with the same random tape and a different oracle, Bob will produce, with non-negligible

probability and in polynomial time, two valid proofs (c_b, D_b) and (c_b^*, D_b^*) with $c_b \neq c_b^* \mod q$ such that $g_1^{D_b} g_b^{c_b} = g_1^{D_b^*} g_b^{c_b^*}$ holds. From this equation, $\mathcal{S}$ can compute $x_b = (D_b^* - D_b)/(c_b - c_b^*) \mod q$. Hence $\mathcal{S}$ can also compute $g_3 = g_1^{x_a x_b} = \alpha^{x_b} = g_b^{x_a}$, even though it does not know x_a. $\mathcal{S}$ then generates $g_{a_2} = g_1^{x_{a_2}}$ for random $x_{a_2} \in \mathbb{Z}_q^*$ and uses Schnorr's protocol to prove knowledge of x_{a_2} (since $\mathcal{S}$ really knows x_{a_2} this is a real proof not a simulated one). Bob then sends g_{b_2} to Alice along with a proof of knowledge (in the random oracle model) of x_{b_2} the discrete logarithm of g_{b_2} to the base g_1. Again, by using the oracle replay attack, $\mathcal{S}$ can find the value x_{b_2}. Let $g_2 = g_1^{x_{a_2} x_{b_2}} = g_{a_2}^{x_{b_2}} = g_{b_2}^{x_{a_2}}$. So at the end of step 1, Alice knows x_b and x_{b_2}.

Step 2. $\mathcal{S}$ randomly selects an element $d \in \mathbb{Z}_q$ and computes $(P_a, Q_a) = (g_3^a, g_1^d)$. So, we have $P_a = g_3^a = g_1^{a x_a x_b} = \beta^{x_b}$. Again, $\mathcal{S}$ can compute this value since it extracted x_b from Bob. Following the definition of g_2, we have: $Q_a = g_1^d = g_1^a g_1^{d-a} = g_1^a g_2^{x_{a_2}^{-1} x_{b_2}^{-1}(d-a)}$. We put $\bar{X} = x_{a_2}^{-1} x_{b_2}^{-1}(d-a)$, where the simulator does not know a. $\mathcal{S}$ sends (P_a, Q_a) to Bob and must also prove that it knows a and $\bar{X}$ satisfying $(P_a, Q_a) = (g_3^a, g_1^a g_2^{\bar{X}})$. Since $\mathcal{S}$ does not know a, the proof required at this step is simulated. In order to produce this proof, $\mathcal{S}$ randomly choose $c, D_1, D_2 \in \mathbb{Z}_q$, and then defines the output of the random oracle on input W_1, W_2 with $W_1 = g_3^{D_1} P_a^c$ and $W_2 = g_1^{D_1} g_2^{D_2} Q_a^c$ to be c (hence $c = h(W_1, W_2)$). With overwhelming probability, Bob has not yet already queried the random oracle at this point. Then $\mathcal{S}$ sends the proof c, D_1, D_2 to Bob. Again *the special honest verifier zero-knowledge* property of the interactive protocol underlying this proof of knowledge ensures that $\mathcal{S}$ produces tuples (c, D_1, D_2) with a distribution indistinguishable from one produced by a real prover knowing a and $\bar{X}$. Next, Bob sends P_b, Q_b to $\mathcal{S}$ along with a proof of knowledge of b, $Y \in \mathbb{Z}_q$ satisfying $P_b = g_3^b$ and $Q_b = g_1^b g_2^Y$. Using the oracle replay attack, $\mathcal{S}$ can find b and Y (note that $\mathcal{S}$ now knows x_b, x_{b_2}, b and Y).

Step 3. In this step, $\mathcal{S}$ and Bob both first compute $(P_a P_b, Q_a Q_b)$, which will be of the form: $(P_a P_b, Q_a Q_b) = (g_3^{a+b}, g_1^{a+b} g_2^{\bar{X}+Y})$. Then $\mathcal{S}$ computes: $R_a = (Q_a Q_b)^{x_a} = g_1^{d x_a} g_1^{b x_a} g_2^{Y x_a} = g_1^{d x_a} g_1^{b x_a} g_1^{x_{a_2} x_{b_2} Y x_a}$. $\mathcal{S}$ can compute this value since it knows d, b, Y, x_{a_2} and x_{b_2}. $\mathcal{S}$ sends R_a to Bob along with a proof that $log_{g_1} \alpha = log_{Q_a Q_b} R_a$. Since $\mathcal{S}$ does not know x_a, the proof required at this step is simulated. To produce this proof, $\mathcal{S}$ randomly chooses $c, D \in \mathbb{Z}_q$. $\mathcal{S}$ then defines the output of the random oracle on input W_1, W_2 with $W_1 = g_1^D \alpha^c$ and $W_2 = (Q_a Q_b)^D R_a^c$ to be c (hence $c = h(W_1, W_2)$). $\mathcal{S}$ then sends the proof (c, D) to Bob. As before, the *special honest verifier zero-knowledge* property of the interactive protocol underlying this proof of knowledge ensures that $\mathcal{S}$ produces tuples (c, D) with a distribution indistinguishable from one produced by a real prover knowing x_a. The simulator's part of the protocol is now complete.

Since the distribution of the simulated views is indistinguishable from that produced by a 'real' Alice (not a simulated one), Bob, after the interaction with $\mathcal{S}$, will be able, as assumed, to find with non-negligible probability $g_2^{\bar{X}}$, which is equal to g_1^{d-a}. Since $\mathcal{S}$ knows d, $\mathcal{S}$ can find $\gamma = g_1^a = g_1^d / g_1^{d-a}$ hence contradicting

the DH assumption. Consequently, such an adversary Bob cannot find $g_2^{\bar{X}}$ hence cannot recover X.

4 Generalization to n Players (Fair Computation of "AND")

Consider the case where n players $J_1, \ldots, J_n$ vote to know if they will play today. A player can answer "yes" (1) if he wants to play or "no" (0) if he does not. We note $X_1, \ldots, X_n$ the answers of the n players. They will play if and only if $X_1 = \ldots = X_n = 1$, i.e. if **all of them** want to play. We wish to find a protocol which:

- convinces all the parties of the correctness of the result.
- does not allow one party whose answer is "no" to get any additional information about the other parties' answers.
- does not allow any party to know how many negative answers there are if the final result is "no".
- is fair, i.e. one party cannot get the result while preventing the others from getting it.

Moreover the scheme will resist "coalition attacks", i.e. there is no interest to actively corrupt some players in order to get some additional information about non-corrupt players.

All the proofs needed are the same as those described in paragraph 2.3.

Step 1:

J_1 randomly chooses $x_1 \in \mathbb{Z}_q^*$ and publishes $y_1 = g_1^{x_1}$.
J_2 randomly chooses $x_2 \in \mathbb{Z}_q^*$ and publishes $y_2 = y_1^{x_2}$.
...
J_n randomly chooses $x_n \in \mathbb{Z}_q^*$ and publishes $y_n = y_{n-1}^{x_n}$.

Every player J_i, $i = 1, \ldots, n$, publishes a proof (Schnorr [23]) that he knows x_i and verifies the other players' proofs. All the players know $g_3 = y_n = g_1^{x_1 \ldots x_n}$.

Step 2: Every player J_i, $i = 1, \ldots, n$, selects a random element $a_i \in \mathbb{Z}_q$ and a random number e_i, $0 \leq e_i < 2^k$, and publishes: $(P_i, Q_i) = (g_3^{a_i} g_0^{e_i}, g_1^{a_i} g_2^{\bar{X}_i})$ and shows that it is correctly formed.

- **Version Without Fairness.** The players set $e_i = 0, i = 1, \ldots, n$ and $P_i = g_3^{a_i}$.
- **Fair Version.** Every player J_i, $i = 1, \ldots, n$, chooses random $a_{i,j} \in \mathbb{Z}_q$ and $e_{i,j} \in \{0, 1\}$, $j = 0, \ldots, k-1$, subject to the condition that $a_i = \sum_{j=0}^{k-1} a_{i,j} 2^j \bmod q$ and $e_i = \sum_{j=0}^{k-1} e_{i,j} 2^j$ and sets $A_{i,j} = g_3^{a_{i,j}} g_0^{e_{i,j}}$, $j = 0, \ldots, k-1$. Then he publishes $A_{i,j}$ with a proof that they are correctly formed. Every player J_i, $i = 1, \ldots, n$, verifies the other players' proofs and checks that $P_l = \prod_{j=0}^{k-1} A_{l,j}^{2^j}$, $l \neq i$, $l = 1, \ldots, n$. All the players know $(P_1, Q_1), \ldots, (P_n, Q_n)$.

Step 3: Every player $J_i, i = 1, \ldots, n$, computes $(P, Q) = (P_1 \ldots P_n, Q_1 \ldots Q_n)$.

J_1 publishes $R_1 = Q^{x_1}$.
J_2 publishes $R_2 = R_1^{x_2}$.
...
J_n publishes $R_n = R_{n-1}^{x_n}$.

Every player $J_i, i = 1, \ldots, n$, publishes a proof that R_i is correctly formed and verifies the other players' proofs ([12]). All the players know $R = R_n = Q^{x_1 \cdots x_n}$.

Step 4:

- **Version Without Fairness.** The players may determine whether $X_1 = \ldots = X_n = 1$ by testing whether $P = R$.
- **Fair Version.** Finally, the players fairly disclose the values of $e_i, i = 1, \ldots, n$, bit by bit without disclose the values of a_i. They publish $(a_{i,j}, e_{i,j})$, for $j = k-1, \ldots, 1$, in turn and verify that $A_{l,j} = g_3^{a_{l,j}} g_0^{e_{l,j}}$, $l \neq i$. Subsequently, they respectively release only $e_{i,0}$ and $f_{i,0}$. Finally, every J_i proves that he knows $log_{g_3} A_{i,0}/g_0^{e_{i,0}}$. This convinces all the players that the bit $e_{i,0}$ is correct. Once the values of $e_i, i = 1, \ldots, n$ are released, the players may determine whether $X_1 = \ldots = X_n = 1$ by testing whether $P = R g_0^{e_1 + \ldots + e_n}$.

Coalitions: Each player receives the same values from one player, each player gives a proof of his constructed values and each player must answer "1" to get the AND of the other players (if he answers "0", the total AND is always "0"). So if m players form an active coalition they will not get more than the AND of the other players (and they get this only if they all answer "1"). So we can say that collusion does not help, i.e. provides no additional information.

5 Fair Computation of Any Binary Operators

- To get a NOR gate, we can use exactly the same algorithm as in section 2, except that in step 2 we take:

$$\begin{cases} Q_a = g_1^a g_2^X \\ Q_b = g_1^b g_2^Y \end{cases}$$

Remark: These values Q_a and Q_b are the same as for the *Socialist Millionaires' Problem* [6] however here we compute in step 3 $P_a P_b$ and $Q_a Q_b$ as for the AND.

In that case, we obtain: $R_{ab} = (Q_a Q_b)^{x_a x_b} = g_3^{a+b} g_2^{(X+Y) x_a x_b}$. Then the test of step 4 is still: $P_a P_b = R_{ab} g_0^{e+f}$? This equality is true if and only if $X + Y = 0$ i.e. if and only if $X = 0$ and $Y = 0$, that is $NOR(X, Y) = 1$.

- To get an $EQUALITY$ gate $(X = Y)$, see the *Socialist Millionaires' Problem* [6]
- For the two following gates $A(X, Y) = AND(X, \bar{Y})$ and $B(X, Y) = AND(\bar{X}, Y)$:

X	Y	$A(X,Y)$
0	0	0
0	1	0
1	0	1
1	1	0

X	Y	$B(X,Y)$
0	0	0
0	1	1
1	0	0
1	1	0

it is sufficient to take :

$$\begin{cases} Q_a = g_1^a g_2^{X-1} \\ Q_b = g_1^b g_2^Y \end{cases} \textit{and} \begin{cases} Q_a = g_1^a g_2^{X+1} \\ Q_b = g_1^b g_2^Y \end{cases} \textit{respectively.}$$

Then : $R_{ab} = \left(\frac{Q_a}{Q_b}\right)^{x_a x_b} = g_3^{a-b} g_2^{(X-Y\pm 1)x_a x_b}$. Then the test is : $P_a P_b = R_{ab} g_0^{e+f}$? For the first case, this equality is true if and only if $X - Y - 1 = 0$, i.e. if and only if $X = 1$ and $Y = 0$, that is $A(X,Y) = 1$ (for the second case $X - Y + 1 = 0$ that is $B(X,Y) = 1$).

– To get a $NAND$ gate it is sufficient to answer "no" at the end of step 4 when $P_a P_b = R_{ab} g_0^{e+f}$ instead of "yes". It is the same thing (we answer "no" instead of "yes" at the end of step 4) with the other gates (OR,XOR, $\neg A$ and $\neg B$).

Remark: Since

$$\begin{aligned} NAND(X,Y) = 1 &\Leftrightarrow X \cdot Y = 0 \\ OR(X,Y) = 1 &\Leftrightarrow \bar{X} \cdot Y = 0 \\ \neg A(X,Y) = 1 &\Leftrightarrow X \cdot \bar{Y} = 0 \\ \neg B(X,Y) = 1 &\Leftrightarrow \bar{X} \cdot \bar{Y} = 0 \end{aligned}$$

we can also use a protocol based on [18] where they compute g^{ab} without revealing a and b. However the scheme of [18] as initially presented is not necessary fair.

Efficiency: The binary operators presented so far can also be computed by Mix and Match techniques [20]. These two techniques use different ideas but for both solutions the basic operations are group exponentiations. With our solutions we need $50 + 4k$ exponentiations (step 1: 8, step 2: $28+2k$, step 3: 12, step 4: $4 + 2(k-1)$) and with Mix and Match, $78 + 4k$ or $96 + 4k$ exponentiations (step 1: 8, step 2: 24, step 3: 36 or 54, step 4: $10 + 4k$).
We can look at the number of rounds too. Our protocol needs $34 + 14k$ rounds (step 1: 6, step 2: $20 + 10k$, step 3: 6, step 4: $2 + 4k$) and the Mix and Match $72+14k$ or $90+14k$ rounds (step 1: 8, step 2: 24, step 3: 36 or 54, step 4: $4+14k$). We think that both solutions are interesting and comparable but the efficiency is slightly better with our solution.

6 Fair Computation of Any Boolean Functions

In this section we rapidly explain how to obtain a fair and efficient algorithm to compute any boolean function (more details are available in the extended

version of this paper). It is sufficient to combine the *Mix and Match* protocol due to M. Jakobsson and A. Juels [20] and *the Socialist Millionaires Problem*'s solution [6].

We suppose that n players wish to compute the output of a binary multiparty function f on secret inputs. The players follow the *Mix and Match* protocol and jointly decrypt fairly the obtained ciphertext value to reveal the output of the function f. To do that, they release bit by bit the values of their private key like in the protocol of the *Socialist Millionaires' Problem* [6] (step 2 and 4).
Like this if one player stops the protocol at any time he will get at most an advantage of one bit (i.e. 50% computations) compared with the other players.

7 Conclusion

In this paper we have shown that the solution of [6] to *The Socialist Millionaires' Problem* can be modified in order to get a fair and efficient solution to compute any boolean function with two inputs without using Mix and Match techniques (i.e. we do not have to crypt all the input table). We first show a solution for the AND (*The Marriage Proposals Problem*) and then to any function of two inputs. Compared with Mix and Match techniques [20], our solution requires to compute less exponentiations ($50 + 4k$ instead of $78 + 4k$ or $96 + 4k$, where k is the security parameter) and the security is directly only related with the Diffie-Hellman assumption (on any group G_q) and not to the Decision Diffie-Hellman assumption. We have also explain how to get a fair computation of any boolean function with any number of players by extending the scheme of [20] (using Mix and Match techniques) in order to obtain fair computations.

References

1. D. Beaver and S. Goldwasser, *Multiparty computation with faulty majority*, Proc. 27th IEEE Symposium on Foundations of Computer Science (FOCS '89), pp 468-473. IEEE Computer Society, 1989.
2. M. Bellare and P. Rogaway, *Random oracles are practical: A paradigm for designing efficient protocols*, ACM CCS '93, pp 62-73, 1993.
3. M. Ben-Or, O. Goldreich, S. Micali and R. L. Rivest, *A fair protocol for signing contracts*, IEEE Trans. on Information Theory, vol 36, pp 40-46. , 1990.
4. B. den Boer, *More Efficient Match-Making and Satisfiability - The Five Card Trick*, Eurocrypt '89, pp 208-217, 1989.
5. D. Boneh and M. Naor, *Timed Commitments*, Crypto '00, pp 236-254, 2000.
6. F. Boudot, B. Schoenmakers and J. Traoré, *A Fair and Efficient Solution to the Socialist Millionaires' Problem*, Journal of Discrete Applied Mathematics, Volume 111, Numbers 1-2, pp 23-36, 2000.
7. G. Brassard and C. Crépeau, *Zero-Knowledge Simulation of Boolean Circuits*, Crypto '86, pp 223-233, 1986.
8. E. Brickell, D. Chaum, I. Damgård and J. van de Graaf, *Gradual and verifiable release of a secret*, Crypto '87, pp 1987.

9. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, *Adaptative security for threshold cryptosystems*, Crypto '99, pp 98-115, 1999.
10. R. Canetti, R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, *The (in)security of distributed key generation in dlog-based cryptosystems*, Eurocrypt '99, pp 295-310, 1999.
11. D. Chaum, *Untraceable electronic mail, return addresses, and digital pseudonyms*, Communications of the ACM, pp 84-88, 1981.
12. D. Chaum and T.P. Pedersen, *Wallet databases with observers*, Crypto '92, pp 89-105, 1993.
13. R. Cramer, I. Damgård and B. Schoenmakers, *Proofs of partial knowledge and simplified design of witness hiding protocols*, Crypto '94, pp 174-187, 1994.
14. C. Crépeau and J. Kilian, *Discreet Solitary Games*, Crypto '93, pp 319-330, 1993.
15. Z. Galil, S. Haber and M. Yung, *Cryptographic Computation: Secure Fault-tolerant Protocols and the Public-Key Model*, Crypto '87, pp 135-155, 1988.
16. T. El Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, Auscrypt '92, pp 244-251, 1992.
17. J. Garay and M. Jakobsson, *Timed Realesed of Standard Digital Signatures*, Proc. Financial Cryptography 2002.
18. R. Gennaro and M. Di Raimondo, *Secure Multiplication of Shared Secrets In The Exponent*, e-print, 2003.
19. S. Goldwasser and L. Levin, *Fair computation of general functions in presence of immoral majority*, Crypto '90, 1990.
20. M. Jakobbson and A. Juels, *Mix and Match: Secure Function Evaluation via Ciphertexts (Extended Abstract)*, Asiacrypt '00, pp 162-177, 2000.
21. B. Pinkas, *Fair Secure Two-Party Computation (Extended Abstract)*, Eurocrypt '03, pp 86-105, 2003.
22. D. Pointcheval and J. Stern, *Security proofs for signature schemes*, Eurocrypt '96, pp 387-398, 1996.
23. C.P. Schnorr, *Efficient signature generation by smart cards*, Journal of Cryptology, pp 161-174, 1991.
24. Syverson, *Weakly Secret Bit Commitments: Applications to Lotteries and Fair Exchanges*, IEEE Computer Security Foundations Workshop, 1998.
25. A. Yao, *Protocols for secure computations*, Proc. 23rd IEEE Symposium on Foundations of Computer Science (FOCS '82), pp 160-164. IEEE Computer Society, 1982.
26. A. Yao, *How to generate and exchange secrets*, Proc. 27th IEEE Symposium on Foundations of Computer Science (FOCS '86), pp 162-167. IEEE Computer Society, 1986.

On the Security of a Certified E-Mail Scheme

Guilin Wang, Feng Bao, and Jianying Zhou

Infocomm Security Department,
Institute for Infocomm Research,
21 Heng Mui Keng Terrace, Singapore 119613
{glwang, baofeng, jyzhou}@i2r.a-star.edu.sg
http://www.i2r.a-star.edu.sg/icsd/

Abstract. As a value-added service for standard e-mail systems, a certified e-mail scheme allows a sender to deliver a message to a receiver in a *fair* way in the sense that either the sender obtains a receipt from the receiver and the receiver accesses the content of the e-mail simultaneously, or neither party gets the expected item. In 2000, Ferrer-Gomila et al. [11] proposed a novel certified e-mail protocol. Their scheme is both efficient and optimistic, since it has only three steps and a trusted third party is not involved in normal cases. Later, Monteiro and Dahab [16] identified an attack on Ferrer-Gomila et al.'s scheme, and further presented a modified scheme. In this paper, we show that their improvement is still insecure by successfully identifying several weaknesses and security flaws. Our attacks also apply to Ferrer-Gomila et al.'s original scheme.

Keywords: certified e-mail, fair exchange, non-repudiation.

1 Introduction

A fair exchange protocol allows two potentially mistrusting parities to exchange digital items over the Internet in a *fair* way, so that either each party gets the other's item, or neither party does. Such protocols include the following different but related variants: non-repudiation protocols [2, 3, 8, 15], e-contract signing protocols [7, 9, 17, 6], certified e-mail schemes [20, 10, 11, 4, 13, 5, 1, 16] etc. For more references and discussions on the relationships between those concepts, please refer to [3, 13, 14].

As a value-added service for standard e-mail systems, a certified e-mail scheme allows a message to be delivered in a fair way between a sender Alice and a receiver Bob, such that *either* Bob accesses the content of the e-mail and Alice obtains a receipt from Bob simultaneously, *or* neither party gets the expected item. In other words, the main purpose of a certified e-mail scheme is to achieve the fair exchange of a message and a receipt. Based on the extent of involvement of the trusted third party (TTP), certified e-mail schemes can be classified into two types: schemes with online TTPs [20, 10, 1] and schemes with offline TTPs [11, 13, 4, 16]. Generally speaking, due to the TTP's participation in every protocol instance, the schemes with online TTPs can be implemented easily, but may

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 48–60, 2004.

be expensive and inefficient, because the TTP may need to be paid and must be involved in each exchange. The schemes with offline TTPs are considered as *optimistic*, since the TTP is not invoked in the execution of the protocol unless one of the two parties misbehaves or the communication channel is out of order.

Abadi et al. proposed an efficient certified e-mail scheme with a light online TTP in [1]. An advantage of their scheme is that users are not required to have PKI certificates. Note that Abadi et al.'s scheme does not provide non-repudiation of origin (NRO), Imamoto and Sakurai [12] proposed an improved scheme that not only meets NRO property but also enables the receiver freely choosing the online TTP or offline TTP.

In [13], Kremer et al. constructed a scheme with a strong security requirement, called *no author-based selective receipt*, in which the sender's identity is not revealed to the receiver unless the receiver issues a receipt to the sender. The TRICERT system [4] is a hybrid scheme that distributes the task of a traditional TTP to less trusted postal agents and a trusted judge who only settles disputes. In addition, based on Micali's certified e-mail scheme [17], Onieva et al. [18] presented a multiparty scheme to delivering a message to a number of receivers.

In 2000, Ferrer-Gomila et al. [11] proposed a novel certified e-mail scheme (FPH scheme, for short). Their scheme is both efficient and optimistic, since it has only three steps and a trusted third party is involved only in exceptions, i.e., one of the two parties is trying to cheat or the communication channel is out of order. Later, Monteiro and Dahab [16] identified an attack on the FPH scheme, and further proposed an improved scheme that was claimed satisfying the following security requirements.

1) **Fairness:** After completion of a protocol run, either each party receives the expected item or neither party receives any useful information about the other's item.
2) **Non-Repudiation:** If an item has been sent from Alice to Bob, Alice cannot deny the origin of the item and Bob cannot deny the receipt of the item.
3) **Timeliness:** At any time during a protocol run, each party can unilaterally choose to terminate the protocol without losing fairness.
4) **Verifiability of Third Party:** If the third party misbehaves, resulting in the loss of fairness for a party, the victim can prove the fact in a dispute.

In this paper, we present a security analysis of Monteiro et al.'s certified e-mail scheme [16] (MD scheme, for short). By successfully identifying several weaknesses and security flaws, we show that the MD scheme is still insecure, since the two most important security requirements of fairness and non-repudiation are not satisfied actually. To demonstrate those security problems concretely and directly, we mount different attacks on the MD scheme. Moreover, similar attacks also apply to the FPH scheme, as the same framework is exploited in those two schemes. At the same time, our attacks illustrate that the executing environment of fair exchange is complicated, so many cares should be paid to design a secure scheme for certified e-mail delivery.

The rest of the paper is organized as follows. In Section 2, notation is introduced. In Section 3, we review the MD scheme, and point out the difference between the MD scheme and the FPH scheme. Then, we present a security analysis of the MD scheme in Section 4. Finally, Section 5 concludes the paper.

2 Notation

In this paper, we use the same notation adopted in [16], which is almost the same as in [11]. For completeness, we list all symbols below.

- A, B and T: The identities of the sender Alice, the receiver Bob, and the TTP.
- X, Y: Concatenation of messages X and Y.
- $H(X)$: Application of a collision-resistant hash function H to message X.
- $PR_i[H(X)]$: The digital signature of party i on message X (which is first hashed with hash function H) by using party i's private key PR_i, where i denotes the identity of any party.
- M: Message delivered to the receiver Bob by the sender Alice.
- K: A secret key for a secure symmetric encryption algorithm.
- "s": A string s.
- $c = E_K(M)$: Ciphertext c produced by encrypting message M under symmetric algorithm E with respect to the secret key K; and decryption of c is denoted by $M = D_K(c)$.
- $k_T = PU_T(K)$: Secret key K is encrypted with the TTP's public key PU_T.
- $h_A = PR_A[H(H(c), k_T)]$: First part of the non-repudiation evidence of origin of message M for Bob generated by Alice.
- $k_A = PR_A[\text{"}key = \text{"}, K]$: Second part of the non-repudiation evidence of origin for Bob generated by Alice.
- $k'_T = PR_T[\text{"}key = \text{"}, K]$: Alternative second part of the non-repudiation evidence of origin for Bob generated by the TTP.
- $\bar{h}_B = PR_B[H(H(c), k_T, h_A)]$: The non-repudiation evidence of receipt of message M for Alice generated by Bob.
- $h_{AT} = PR_A[H(H(c), k_T, h_A)]$: An evidence that Alice has requested the TTP's intervention.
- $\bar{h}_{BT} = PR_B[H(H(c), k_T, h_A, \bar{h}_B)]$: An evidence that Bob has requested the TTP's intervention.
- $\bar{h}'_B = PR_T[H(\bar{h}_B)]$: The TTP's signature on $\bar{h}_B$ that shows its intervention.

3 The MD Scheme

In this section, we first overview the MD scheme proposed by Monteiro and Dahab in [16], which is an improvement of the FPH scheme [11]. After that, we briefly point out the difference between the FPH scheme and the MD scheme, and discuss Monteiro and Dahab's attack on the FPH scheme.

Like all certified e-mail schemes, the MD scheme also consists of a dispute resolution policy, and three sub-protocols, i.e., the exchange protocol, the cancel protocol, and the finish protocol. The *dispute resolution policy* defines the evidences of non-repudiation of origin (NRO) and non-repudiation of receipt (NRR), and the procedures how a judge settles potential disputes over NRO or NRR between different parties. The *exchange protocol* is the main protocol, which is executed jointly by the sender Alice and the receiver Bob in *normal situation*, i.e., both involved parties behave honestly according to the protocol specification and the communication channel is in order. However, as we mentioned before, the difficulty is to meet security requirements even in *abnormal situation* where one party is trying to cheat the other or the communication channel is out of service. As pointed out by Micali [17], fair exchange protocols have to solve an essential unfair problem in a fair way. Therefore, in abnormal situations, the *cancel protocol* and the *finish protocol* are designed to achieve fairness for the sender Alice and the receiver Bob, respectively, under the help of the TTP. More specifically, the cancel protocol allows the sender Alice to cancel a protocol execution in the following situations: (1) Bob does not respond; (2) Bob does not respond correctly or timely; or (3) The communication channel interrupts. Analogously, the finish protocol protects the receiver Bob from the cheating of Alice or the failure of communications.

We now review in detail the three sub-protocols and the dispute resolution policy of the MD scheme [16].

3.1 Description of Sub-protocols

(1) **The Exchange Protocol.** Assume that Alice wants to deliver a message M to the receiver Bob with a guarantee that Bob can access the message M if and only if she obtains a receipt from Bob. To this end, the sender Alice and the receiver Bob run the following exchange protocol jointly.

$$\begin{array}{lll} \text{(e1).} & A \longrightarrow B: & c, k_T, h_A \\ \text{(e2).} & B \longrightarrow A: & \bar{h}_B \\ \text{(e3).} & A \longrightarrow B: & k_A \end{array} \tag{1}$$

That is, in normal case, Alice first selects a symmetric encryption key K, and then encrypts the secret key K and message M by computing $k_T = PU_T(K)$ and $c = E_K(M)$. Then, she generates her signature h_A for message $(H(c), k_T)$ to show that c and k_T are originated from herself, instead of others. Finally, Alice sends (c, k_T, h_A) to the receiver Bob as message flow (e1). After validating that h_A is Alice's signature on message $(H(c), k_T)$, Bob generates and returns his signature $\bar{h}_B$ to show that he has received the ciphertext c, the encrypted key k_T, and Alice's signature h_A on $(H(c), k_T)$. If $\bar{h}_B$ is correct, Alice sends Bob k_A as message flow (e3) to reveal the secret key K.

Therefore, after the successful completion of the exchange protocol, Alice holds the NRR evidence $\bar{h}_B$, and Bob obtains the NRO evidence (h_A, k_A) and then can access message M by decrypting c with secret key K, i.e., $M = D_K(c)$. Note that the authors of [11] pointed out that the RSA cryptosystem [19] is

specially suitable to their scheme, so Bob can get the value of the secret key K from k_A or k'_T by using the public key of Alice or the TTP, respectively.

However, if the exchange protocol is not executed successfully, the resulting situation may be unfair for one of involved parties. In such cases, either Alice or Bob, or both of them, can initiate the cancel protocol or the finish protocol, respectively, to achieve fairness under the help of the TTP.

(2) **The Cancel Protocol.** If the sender Alice does not receive the expected value of $\bar{h}_B$ correctly or timely, she can execute the following cancel protocol with the TTP and then abort the protocol instance with the receiver Bob.

$$\begin{array}{lll} \text{(c1).} & A \longrightarrow T: & H(c), k_T, h_A, h_{AT} \\ \text{(c2).} & \textbf{If } (finished = true),\ T \text{ retrieves } \bar{h}_B & \\ & T \longrightarrow A: & \bar{h}_B, \bar{h}'_B \\ & \textbf{Else } T \text{ stores } (cancelled = true) & \\ & T \longrightarrow A: & PR_T[H(\text{“}cancelled\text{”}, h_A)] \end{array} \tag{2}$$

We explain the cancel protocol in detail as follows. To abort a protocol execution, Alice first sends $(H(c), k_T, h_A, h_{AT})$ as a cancel request to the TTP. Then, the TTP verifies the correctness of message flow (c1), i.e., whether h_A and h_{AT} are Alice's signatures on messages $(H(c), k_T)$ and $(H(c), k_T, h_A)$, respectively. If this is not the case, an error message is sent to Alice. Otherwise, the TTP proceeds in one of two possible ways according to the values of variables *cancelled* and $finished$ (their default values are set as $false$). If $finished = true$, the TTP retrieves the stored NRR evidence $\bar{h}_B$, and sends it to Alice together with a token $\bar{h}'_B$ to prove its intervention, since $finished = true$ means that Bob has obtained k'_T (and then the secret key K) from the TTP by successfully executing the finish protocol (see below). On the other hand, if Bob has not contacted the TTP previously, i.e., $finished = false$, the TTP sends Alice a cancellation token $PR_T[H(\text{“}cancelled\text{”}, h_A)$ to abort the transaction, and stores the status variable as $cancelled = true$ to satisfy future possible petitions from Bob.

(3) **The Finish Protocol.** Similarly, if the receiver Bob has sent NRR evidence $\bar{h}_B$ to the sender Alice but does not receive the expected value k_A from Alice correctly or timely, he can get the alternative second part NRO evidence k'_T (and then the secret key K) from the TTP by running the following finish protocol.

$$\begin{array}{lll} \text{(f1).} & B \longrightarrow T: & H(c), k_T, h_A, \bar{h}_B, \bar{h}_{BT} \\ \text{(f2).} & \textbf{If } (cancelled = true) & \\ & T \longrightarrow B: & PR_T[H(\text{“}cancelled\text{”}, \bar{h}_B)] \\ & \textbf{Else } T \text{ stores } (finished = true) \text{ and } \bar{h}_B & \\ & T \longrightarrow B: & k'_T \end{array} \tag{3}$$

In more details, to get the value of k'_T from the TTP directly, Bob sends message flow (f1) as his resolution request to the TTP, where Bob's signature $\bar{h}_{BT}$ shows that this request comes from Bob instead of other users. Upon receiving Bob's request, the TTP checks the correctness of $(H(c), k_T, h_A, \bar{h}_B, \bar{h}_{BT})$. If it is not the case, the TTP sends an error message to Bob. Otherwise, it proceeds in

one of two possible ways according to the values of status variables. If $cancelled = true$, the TTP sends Bob a cancellation token $PR_T[H(\text{“}cancelled\text{”}, \bar{h}_B)]$ to exempt his liability for sending $\bar{h}_B$, since $cancelled = true$ means that Alice has already aborted the transaction by successfully executing the cancel protocol (see above) with the TTP, and that the TTP has given a cancellation token to Alice. If $cancelled = false$, i.e., Alice has not cancelled the transaction with the TTP, the TTP first gets the secret key K by decrypting k_T. After that, it generates and sends k'_T to Bob so that he is able to derive the value of K and then access the content of the encrypted message c by computing $M = D_K(c)$. At the same time, the TTP stores the NRR token $\bar{h}_B$ from Bob, and the status variable as $finished = true$ to satisfy future possible petitions from Alice.

3.2 Dispute Resolution Policy

In some day after the completion of a protocol execution (with or without the TTP's participation), it may be necessary to handle the following two types of dispute resolution requests:

- **Repudiation of Origin:** Bob claims having received message M from Alice but Alice denies having sent message M to Bob.
- **Repudiation of Receipt:** Alice claims having sent message M to Bob but Bob denies having received message M from Alice.

An external judge should resolve these disputes according the dispute resolution policy specified as follows.

- **Resolving Repudiation of Origin.** As a resolution request, Bob first sends the judge M, c, k_T, h_A, and k_A or k'_T. Then, the judge verifies the validity of signatures h_A, k_A or k'_T, and whether $D_K(c) = M$, where the secret key K is derived from k_A or k'_T. If any of the above checks fails, the judge dismisses Bob's claim. Otherwise, i.e., Bob's evidence is correct, the judge further asks whether Alice holds the cancellation token $PR_T[H(\text{“}cancelled\text{”}, h_A)]$. If the correct token is provided, then the judge concludes that the TTP has acted mistakenly. In other situation, i.e., Bob has correct NRO evidence and Alice cannot provide correct cancellation token, the judge rules in favor of Bob, i.e., the verdict is that Alice indeed sent message M to Bob.
- **Resolving Repudiation of Receipt.** Similarly, to get the judge's jurisdiction for a dispute over receipt, Alice submits $(M, c, k_T, h_A, \bar{h}_B, K)$ to the judge. Upon receiving Alice's request message, the judge verifies the validity of signatures $h_A, \bar{h}_B$, whether k_T is the encryption of K under the public key of the TTP. If any of these checks is incorrect, then the judge dismisses Alice's claim. However, positive answers for those checks are not sufficient to guarantee that Alice's claim is true. The judge should further require Bob to provide a cancellation token $PR_T[H(\text{“}cancelled\text{”}, \bar{h}_B)]$. If Bob is able to provide it, this means Alice is trying to cheat, so the judge rejects Alice's request and solves the dispute in Bob's favor, i.e., Bob has not received message M from Alice. If Bob cannot provide that token, the judge will check whether

$M = D_K(c)$. If this final check is positive, the judge will side with Alice, i.e., Bob received M from Alice. Otherwise, if the previous check fails, the judge rejects Alice's request. Finally, if Alice's NRR evidence is correct and Bob is able to provide a correct cancellation token $PR_T[H(\text{“}cancelled\text{”}, \bar{h}_B)]$, then the judge concludes that the TTP has acted improperly.

3.3 The FPH Protocol and Its Security

As noted by the authors of [16], the FPH scheme and the MD scheme are almost the same, except in the former $\bar{h}_B$, $\bar{h}'_B$ and $\bar{h}_{BT}$ are replaced by h_B, h'_B and h_{BT}, respectively, where

$$\begin{aligned} h_B &= PR_B[H(H(c), k_T)], \\ h'_B &= PR_T[H(h_B)], \\ h_{BT} &= PR_B[H(H(c), k_T, h_A, h_B)]. \end{aligned} \tag{4}$$

Consequently, in the FPH scheme, the NRO evidence h_A and the NRR evidence h_B have a symmetric structure. Based on this observation, Monteiro and Dahab [16] pointed out an attack against the FPH scheme. In their attack, the receiver initiates both a finish protocol and a cancel protocol by using the same public information of a protocol instance executed with the sender Alice. Here, we do not review their attack in detail, but we want to point out that the success of the attack depends on the following assumption: The TTP does not check whether $(h(c), k_T)$ are used repeatedly in different instances of the cancel and finish protocol. If this check is added, their attack is thwarted (though the FPH scheme is indeed insecure, as we will see later).

Moreover, to avoid their attack, Monteiro and Dahab introduced asymmetric structures for NRO and NRR evidences, and then obtained an improved scheme, i.e., the MD scheme, which is reviewed in previous sections. In the next section, we will show that the MD scheme is still insecure, though they claimed their scheme is secure.

4 Security of the MD Scheme

In [11, 16], the authors provided security analysis in detail to show that their schemes are secure, i.e., satisfying all desirable security properties. However, contrary to their claim, we identify several weaknesses and security flaws in their schemes. To illuminate those security problems concretely and directly, we demonstrate different attacks in this section. For simplicity, we only describe the attacks on the MD scheme. Similar attacks also apply to the FPH scheme, as the same framework is exploited in those two schemes. In our discussion, we not only describe the details about the attacks, but also try to find the reasons why the MD is still insecure.

4.1 On the Communication Channels

We note that the authors of [11, 16] did not explicitly specify what kinds of communication channels are needed in their schemes. Actually, only [11] pointed out

that cryptographic operations to achieve confidentiality are omitted in order to simplify the explanation. However, as practice-oriented systems, certified e-mail schemes are expected to involve as few as possible cryptographic operations while the desirable security requirements are still satisfied. So, assumptions on the communication channels should be specified clearly in certified e-mail schemes (and maybe almost all security protocols). Otherwise, misunderstandings may occur easily in the system evaluation and/or implementation procedures.

Here, we emphasize that secure channels in some sense are necessary in both the MD scheme and the FPH scheme. For example, as a signature with message recovery, the value of k_A and k'_T must be protected under a secure encryption scheme. Otherwise, an eavesdropper Eve can derive the message M which is delivered in encrypted manner between Alice and Bob. That is, if k_A and k'_T are sent to Bob straightforwardly without additional protection, a passive attacker Eve may first intercept the ciphertext c and the encrypted secret key k_A or k'_T, and then get message M by computing $M = D_K(c)$, where the secret key K is recovered from k_A or k'_T. Therefore, in both schemes k_A and k'_T must be protected in appropriate ways, if the confidentiality of delivered message M is supposed to provide.

In the following discussion, we assume that there exist authenticated and confidential channels among the sender Alice, the receiver Bob and the TTP. That is, all message flows communicated between any two parties of them cannot be impersonated, modified, inserted or decrypted by a third party. Such channels are usually called as *secure*. Later analysis will show that both the MD scheme and the FPH scheme are insecure, even in the environments where secure channels are provided.

4.2 On the Dispute Resolution Policy

Dispute resolution policy plays a very important role in all certified e-mail schemes (and non-repudiation protocols). That is, a judge settles all NRO or NRR disputes according to the specification of dispute resolution policy exactly. With a secure dispute resolution policy, any dishonest party cannot cheat other parties except a negligible probability. However, we find that according to the specification of the MD protocol, the sender Alice can maliciously frame the TTP even though the TTP acts honestly.

In the specification of NRO resolution protocol, even though Bob provided the correct NRO evidence, Alice will be judged as innocent if she can provide the cancellation token $PR_T[H(\text{“}cancelled\text{”}, h_A)]$. Therefore, to frame the TTP dishonest sender Alice may mount the following attack: After running the exchange protocol successfully with Bob, Alice dishonestly initiates the cancel protocol with the TTP to get a cancellation token. In more details, Alice first correctly prepares and sends (c, k_T, h_A) to the receiver Bob as message flow (e1). Upon receiving Bob's signature $\bar{h}_B$ via message flow (e2), Alice returns k_A to Bob as message flow (e3). Up to this point, a successful protocol execution is completed. However, malicious Alice may initiate the cancel protocol with the TTP by sending out $(H(c), k_T, h_A, h_{AT})$ as message flow (c1). Assumed that Bob has

not executed the finish protocol with the TTP, Alice will obtain the cancellation token $PR_T[H(\text{“}cancelled\text{”}, h_A)]$ from the TTP via message flow (c2).

After the above procedures have completed, honest receiver Bob (mistakenly) believes that the protocol is executed successfully since he sent $\bar{h}_B$ to Alice and received correct NRO evidence (h_A, k_A). However, when Bob wants to prove that Alice has sent message M to him, Alice can successfully deny having sent message M to Bob by providing the cancellation token $PR_T[H(\text{“}cancelled\text{”}, h_A)]$. According to the specification of the dispute resolution policy, the judge will conclude that the TTP has acted improperly. So the result is that the honest TTP is framed. In the real world, the TTP may have to take some responsibilities for his faults arbitrated by the judge, for example compensating an amount of money to Alice and Bob.

Actually, the fact of Bob holding (h_A, k_A) and Alice being able to provide $PR_T[H(\text{“}cancelled\text{”}, h_A)]$ has already implied Alice's dishonest behavior. Because if Alice is honest, there are only two possible ways she has done: (1) Obtaining $PR_T[H(\text{“}cancelled\text{”}, h_A)]$ by running the cancel protocol but not revealing k_A to Bob; or (2) Sending k_A to Bob without running the cancel protocol (and so having no $PR_T[H(\text{“}cancelled\text{”}, h_A)]$). The existence of the above attack means that in the MD scheme, the NRO dispute resolution policy is illogical. In other words, if Bob provided correct (h_A, k_A), Alice should be judged as the originator of message M, regardless whether she holds the cancellation token. On the other hand, the judge will still conclude that the TTP has acted improperly if Bob submitted correct NRO evidence (h_A, k'_T) and Alice provided valid $PR_T[H(\text{“}cancelled\text{”}, h_A)]$.

In addition, note that the NRO evidence (h_A, k_A) or (h_A, k'_T) can only be explained as “Alice indeed sent message M to *somebody*”, instead of “Alice indeed sent message M to *Bob*”. The reason is that after receiving Bob's valid NRO dispute request, (M, c, k_T, h_A, k_A) or (M, c, k_T, h_A, k'_T), the judge actually cannot recognize whether Bob is the receiver designated by Alice, since those tuples do not include any information about Bob's identity.

4.3 Cheating from the Receiver

The main purpose of the finish protocol is to protect honest receiver Bob, because the sender Alice may dishonestly refuse to reveal the value of k_A to Bob after she has obtained NRR evidence $\bar{h}_B$ from Bob via message follow (e2). If Alice really cheated like that, Bob runs the finish protocol to get k'_T from the TTP as a substitute for k_A, i.e., the second part of NRO evidence. However, the TTP needs to store the value of $\bar{h}_B$ and the status value *cancelled* and $finished$, because the TTP has to prevent dishonest Bob from trying to get k'_T from itself directly without sending the NRR evidence $\bar{h}_B$ to Alice. However, the goal of fairness is not achieved.

In this section, we demonstrate an attack that allows a dishonest receiver Bob to access the encrypted message m without revealing the receipt $\bar{h}_B$ to anybody, including the sender Alice and the TTP. In this attack, we assume that Bob colludes with his friend A' to get the secret key K from the TTP. The attack is illuminated as follows:

$$
\begin{array}{lll}
\text{(e1).} & A \longrightarrow B: & c, k_T, h_A \\
\text{(1').} & B \longrightarrow A': & k_T \\
\text{(2').} & A' \longrightarrow B: & c_0, h_{A'} \\
\text{(f1').} & B \longrightarrow T: & H(c_0), k_T, h_{A'}, \bar{h}_{B0}, \bar{h}_{BT0} \\
\text{(f2').} & \textbf{If } (cancelled = true) & \\
 & T \longrightarrow B: & PR_T[H(\text{“}cancelled\text{”}, \bar{h}_{B0})] \\
 & \textbf{Else } T \text{ stores } (finished = true) \text{ and } \bar{h}_{B0} & \\
 & T \longrightarrow B: & k'_T
\end{array}
\tag{5}
$$

Now, we explain the above attack in detail. When dishonest receiver Bob obtains (c, k_T, h_A) via message flow (e1), he interrupts the protocol execution permanently. In order to obtain k'_T (and then the secret key K) from the TTP, Bob sends k_T to his friend A'. Then, A' selects a random number c_0, signs the message $(H(c_0), k_T)$ by computing $h_{A'} = PR_{A'}[H(H(c_0), k_T)]$, and returns back $(c_0, h_{A'})$ to Bob. Upon receiving $(c_0, h_{A'})$, Bob generates his signatures $\bar{h}_{B0} = PR_B[H(H(c_0), k_T, h_{A'})]$ and $\bar{h}_{BT0} = PR_B[H(H(c_0), k_T, h_{A'}, \bar{h}_{B0})]$. After that, Bob sends $(H(c_0), k_T, h_{A'}, \bar{h}_{B0}, \bar{h}_{BT0})$ as message flow (f1') to the TTP by initiating the finish protocol. Since this message flow is indeed valid, the TTP believes that A' and Bob are running the protocol, and that Bob wants to finish this protocol instance now. So the TTP decrypts k_T to get K, then generates and sends k'_T to Bob. Therefore, under the help of A' and the TTP, Bob can read the content of encrypted e-mail c by decrypting $M = D_K(c)$, while the honest sender Alice does not obtain the receipt $\bar{h}_B$ from Bob. Furthermore, if necessary, Bob can prove that Alice has sent message M to him by providing NRO evidence (h_A, k'_T). Therefore, this attack shows that the MD scheme is unfair for the sender.

The above attack employs the following weakness in the MD protocol: the identities of the sender and the receiver are not embedded in k_T. That is, when the TTP derives the secret key K by decrypting k_T, it does not know who is the originator of this encrypted key and who is the expected receiver. Consequently, the TTP cannot determine whether the dispute resolution requesters are the parties involved in the protocol instance in which k_T is generated. Furthermore, note that the TTP cannot check whether c corresponds with a meaningful plaintext under the secret K derived from k_T, because only the hash value $H(c)$ instead of c is provided to it in both the cancel protocol and the finish protocol. In this way, as pointed out by Abadi et al. [1], the privacy and efficiency are guaranteed better: (1) Without the ciphertext c, even the TTP does not know the delivered message M between Alice and Bob; and (2) The overhead of communications between the TTP and users is proportional to the number of dispute resolution requests rather than the length of message exchanged among users. However, the point is that security is compromised.

Remark 1. The above attack is significantly stronger than Monteiro et al.'s attack on the FPH scheme. Namely, our attack is independent of the specific checking procedures exploited by the TTP in the cancel and finish protocol, since the

message flow (f1') sent to the TTP is a totally new message for the TTP. However, Monteiro et al.'s attack relies on the assumption that the TTP does not check the repeated usage of the same information $(H(c), k_T)$, as we mentioned before.

Remark 2. If both the ciphertext c and the encrypted secret k_T are transferred via a public channel, it is not difficult to see that an eavesdropper B' colluding with his friend A' can also mount the above attack. By doing so, B' could know the content M of the ciphertext c, though M is delivered between Alice and Bob in an encrypted way.

4.4 Other Weaknesses in the MD Scheme

Actually, the MD scheme has some other weaknesses. The first one is that the TTP has to store a lot of messages in its database. Specifically, for all successful dispute resolution requests, the TTP has to store the values of status variables, related signatures etc. Correctly maintaining the status variables' values are very important, since the variables *cancelled* and $finished$ are exclusive to each other. If the values of such variables are lost or confused, a special protocol instance may be successfully cancelled by the sender Alice and be successfully finished by the receiver Bob simultaneously. Then, the TTP may issue inconsistent tokens and has to bear potential liability in the future. We remark that the technique of cut-off time introduced in [17] can be used to address this problem in a great degree. That is, let k_T, h_A, and $\bar{h}_B$ include the same cut-off time t. After the specified cut-off time t, the TTP neither stores the messages nor responds dispute resolution requests related with t. However, cares should be paid for the possible attacks resulting from the drift of clocks among different parties.

Another weakness in the MD scheme is that to thwart both the cancel protocol and the finish protocol being successfully executed for the same protocol instance, what are the procedures the TTP should follow? For example, only checking whether $H(c)$ is repeated or checking all message components of dispute requests. In [11, 16], such procedures are not provided. Based on different implementation, specific attacks may be mounted.

5 Conclusion

This paper presented a security analysis of the certified e-mail scheme proposed in [16], which is an improvement of the scheme in [11]. Our analysis showed that those two certified e-mail schemes are all insecure. The most serious problem is that a receiver can access the content of encrypted e-mail message without issuing a receipt. Therefore, both of those two schemes are actually unfair, contrary to the original authors's claims. The attacks identified in this paper also illustrated that the executing environment of certified e-mail delivery is complicated, so many details need to be considered in the design of a secure scheme. For example, clear assumption on communication channels, correct definition and explanation on the non-repudiation evidences, detailed specification on the

TTP's action, necessary link information among different message flows, explicit identity information on related parties etc. In the future research, we will consider to design a new certified e-mail scheme.

Acknowledgements. The authors would like to thank the anonymous referees for their helpful and detailed suggestions on the improvement of this paper.

References

1. M. Abadi, N. Glew, B. Horne, and B. Pinkas. Certified email with a light on-line trusted third party: Design and implementation. In: *Proc. of 2002 International World Wide Web Conference* (*WWW'02*), pp. 387-395. ACM press, 2002.
2. N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In: *Proc. of AMC Conference on Computer and Communications Security* (*CCS'97*), pp. 7-17. ACM Press, 1997.
3. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18 (4): 591-606, 2000.
4. G. Ateniese, B. de Medeiros, and M.T. Goodrich. TRICERT: A distributed certified E-mail scheme. In: *Proc. of Symposium on Network and Distributed Systems Security* (*NDSS'01*). Internet Society, 2001.
5. G. Ateniese and C. Nita-Rotaru. Stateless-receipient certified E-mail system based on verifiable encryption. In: *CT-RSA'02*, LNCS 2271, pp. 182-199. Springer-Verlag, 2002.
6. F. Bao, G. Wang, J. Zhou, and H. Zhu. Analysis and improvement of Micali's fair contract signing protocol. In: *Information Security and Privacy* (*ACISP'04*), LNCS 3108, pp. 176-187. Springer-Verlag, 2004.
7. M. Ben-Or, O. Goldreich, S. Micali, and R. L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1): 40-46, 1990.
8. C. Boyd and P. Kearney. Exploring fair exchange protocols using specification animation. In: *Information Security Workshop* (*ISW'00*), LNCS 1975, pp. 209-223. Springer-Verlag, 2000.
9. I.B. Damgård. Practical and provably secure release of a secret and exchange of signatures. *Journal of Cryptology*, 8(4): 201-222, 1995.
10. R. Deng, L. Gong, A. Lazar, and W. Wang. Practical protocol for certified electronic mail. *Journal of Network and Systems Management*, 1996, 4(3):279-297.
11. J. L. Ferrer-Gomila, M. Payeras-Capella, and L. Huguet-Rotger. An efficient protocol for certified elctronic mail. In: *Information Security Workshop* (*ISW'00*), LNCS 1975, pp. 237-248. Springer-Verlag, 2000.
12. K. Imamoto and K. Sakurai. A cerified e-mail system with reciever's selctive usage of delivery authortiy. In: *Indocrypt 2002*, LNCS 2551, pp. 326-338. Springer-Verlag, 2002.
13. S. Kremer and O. Markowitch. Selective receipt in cerified e-mail. In: *Indocrypt 2001*, LNCS 2247, pp. 136-148. Springer-Verlag, 2001.
14. S. Kremer, O. Markowitch, and J. Zhou. An intensive survey of fair non-repudiation protocols. *Computer Communications*, 25(17): 1606-1621. Elsevier, Nov. 2002.
15. S. Gurgens, C. Rudolph, and H. Vogt. On the security of fair non-repudiation protocols. In: *Information Security Conference* (*ISC'03*), LNCS 2851, pp. 193-207. Springer-Verlag, 2003.

16. J.R.M. Monteiro and R. Dahab. An attack on a protocol for certified delivery. In: *Information Security Conference* (*ISC'02*), LNCS 2433, pp. 428-436. Springer-Verlag, 2002.
17. S. Micali. Simple and fast optimistic protocols for fair electronic exchange. In: *Proc. of 22th Annual ACM Symp. on Principles of Distributed Computing* (*PODC'03*), pp. 12-19. ACM Press, 2003.
18. J.A. Onieva, J. Zhou, and J. Lopez. Enhancing certied email service for timeliness and multicasting. In: *Proc. of 4th International Network Conference* (*INC'04*), pp. 327-336, Plymouth, UK, July 2004.
19. R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, Feb. 1978, 21(2): 120-126.
20. J. Zhou and D. Gollmann. Certified electronic mail. In: *Computer Security - ESORICS'96*, LNCS 1146, pp. 160-171. Springer-Verlag, 1996.

Multiplicative Homomorphic E-Voting*

Kun Peng[1], Riza Aditya[1], Colin Boyd[1], Ed Dawson[1], and Byoungcheon Lee[1,2]

[1] Information Security Research Centre,
IT Faculty, Queensland University of Technology
{k.peng, c.boyd, e.dawson, r.aditya}@qut.edu.au
http://www.isrc.qut.edu.au
[2] Joongbu University, Korea
sultan@joongbu.ac.kr

Abstract. All the currently existing homomorphic e-voting schemes are based on additive homomorphism. In this paper a new e-voting scheme based on multiplicative homomorphism is proposed. In the tallying phase, a decryption is performed to recover the product of the votes, instead of the sum of them (as in the additive homomorphic e-voting schemes). Then, the product is factorized to recover the votes. The new e-voting scheme is more efficient than the additive homomorphic e-voting schemes and more efficient than other voting schemes when the number of candidates is small. Strong vote privacy and public verifiability are obtained in the new e-voting scheme.

1 Introduction

Two main methods have been applied to design e-voting schemes: mix network and homomorphic tallying. Both methods can protect vote privacy when threshold trust is assumed. In regard to efficiency, it is demonstrated in [2] that mix network is more suitable for elections with a large number of candidates or choices (e.g. preferential voting) and homomorphic tallying is more suitable for elections with a small number of candidates or choices (e.g. "YES/NO" voting) as the latter's cost is linear in the number of candidates or choices.

Current homomorphic e-voting schemes employ an additive homomorphic encryption algorithm (e.g. Paillier encryption) to encrypt the votes and exploit additive homomorphism of the encryption algorithm to recover the sum of votes for any candidate or choice with a single decryption. As no single vote is decrypted, vote privacy is protected. It is surprising that multiplicative homomorphism has never been employed to design any voting scheme, although it may lead to better performance.

The contribution of this paper is a design of a multiplicative homomorphic voting scheme. In a multiplicative homomorphic voting scheme, a multiplicative

* The research in this paper was supported by Australian Research Grants DPO345458 and LX0346868.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 61–72, 2004.

homomorphic encryption algorithm (e.g. ElGamal encryption) to encrypt the votes and a single decryption is performed to calculate the product of votes. Then the product is factorized and the votes are recovered. Like in additive homomorphic voting, no single vote is decrypted in multiplicative homomorphic voting, so vote privacy is protected too. The most important advantage of multiplicative homomorphic voting is that it is always more efficient than additive homomorphic voting and more efficient than other voting schemes when the number of candidates is small. In brief, multiplicative homomorphic voting improves efficiency without compromising vote privacy or public verifiability.

2 Related Work

In an election, the voters select a certain number of winners from a few candidates. At first the identities of the candidates and the number of expected winners are declared. Then every bidder appoints some candidates in his bid, whose number is equal to the number of expected winners. Finally some talliers count the votes and declare the voting result. In an e-voting system, tallying must be performed without revealing any vote.

Definition 1. *If after the voting every vote is only known to distribute uniformly in the vote space (containing all the possible choices), we say that* ***complete vote privacy*** *is achieved. If after the voting every voter's choice is only known to be among a large number of published votes, whose number is much larger than the number of possible choices, we ay that* ***strong vote privacy*** *is achieved.*

So far, two methods have been employed to protect vote privacy in voting schemes. The first one is mix network. In voting schemes using this method [15, 34, 28, 31, 32, 27, 20, 7, 17, 36, 1, 23, 9] (called mix voting), the votes are shuffled in the mix network and then decrypted separately. Although every single vote is decrypted, they cannot be linked to the voters after being shuffled. So, vote privacy is achieved. The second is homomorphic tallying, which exploits the homomorphism of the encryption algorithm (used to encrypt the votes) to implement the tallying without decrypting any single vote. E-voting schemes employing the second method are called homomorphic voting and include [18, 5, 33, 6, 11, 12, 3, 26, 35, 19, 4, 21, 13, 22, 24, 25]. Since no single vote is decrypted, vote privacy is obtained. Homomorphic voting schemes are efficient when the number of candidates or choices is small. However, homomorphic voting has a drawback: each vote must be verified to be valid. Without the vote validity check, correctness of the tallying cannot be guaranteed. When the number of candidates or choices is large (e.g. in a preferential voting), computational and communicational cost for the proof and verification of vote validity is so high that homomorphic voting becomes less efficient than mix voting. So, it is widely believed that homomorphic voting is only suitable for elections with a small number of candidates or choices (e.g. "YES/NO" voting).

An encryption scheme is additive homomorphic if $E(m_1+m_2) = E(m_1)E(m_2)$ for any messages m_1 and m_2 where $E()$ stands for the encryption function. In an

additive homomorphic voting scheme, each bidder makes a choice for every candidate (1 for the candidate or 0 against the candidate), encrypts his choices as his encrypted vote. Then he proves that his vote is valid, namely every choice encrypts 0 or 1 and the number of 1s encrypted in his vote is equal to the expected number of winners specified in the voting rule. The talliers verify that each vote is valid. Then they decrypt the product of encrypted choices for each candidate or the product of all the encrypted votes (in some special voting schemes [24, 25] each voter combines his choices for all the candidates in one ciphertext) to find out the sum of votes for each candidate without decrypting any single vote.

One of two possible additive homomorphic encryption algorithm are usually employed: Paillier encryption or modified ElGamal encryption. Paillier encryption is inherently additive homomorphic and more frequently applied. The original ElGamal encryption scheme can be simply modified to be additive homomorphic: a message is used as an exponent in an exponentiation computation, then the exponentiation is encrypted using the original ElGamal encryption. A passive result of this modification is that a search for logarithm must be performed in the decryption function, which becomes inefficient when the searching space is not too small. The modified ElGamal encryption is employed in homomorphic voting schemes [18, 22, 24, 25], where the details of the modification and the consequent search are described in detail.

A disadvantage of additive homomorphic voting compared to multiplicative homomorphic voting is inefficiency due to the following reasons.

- If Paillier encryption is employed, the following drawbacks in efficiency exist.
  - Inefficient set-up
    In voting schemes, the private key of the encryption algorithm must be generated and shared by multiple talliers, so that it is not needed to trust any single party to achieve vote privacy. As the private key is a factorization secret in Paillier encryption, distributed key generation is highly inefficient. In comparison, distributed key generation in ElGamal (distributed generation of a secret logarithm as the private key) is much more efficient as described in [14, 30, 16].
  - Multiple encryption
    Usually, a voter has to perform an encryption for each candidate and prove each of his encryptions contains a valid message.
  - Inefficiency of multiplicative and exponentiation computations
    In Paillier encryption, each multiplication is performed modulo N^2 where N is the product of two large primes and its factorization is the private key (see [29] for details). In comparison, in original ElGamal encryption, each multiplication is performed modulo p, a large prime. If the same security strength is required, N and p should have the same length (e.g. 1024 bits). As the modulus in Paillier encryption scheme is a square and usually the computation for modular multiplication is quadratic in the operand size, multiplication in Paillier encryption scheme is more costly than that in ElGamal encryption scheme. Although Chinese Remainder Theorem can be employed to improve the efficiency of multiplicative

computation with a composite modulus in Paillier encryption scheme, Paillier admitted this efficiency improvement is only available in key generation and decryption when the factorization of N is known. Paillier indicated that a multiplication in Paillier encryption is more than three times as costly as a multiplication in ElGamal encryption when N and p should have the same length (e.g. 1024 bits). Usually distributed decryption is employed in voting schemes to minimize trust and strengthen robustness, so the factorization of N is not known to any single tallier, who performs the decryption. Therefore, we can assume that when the same security strength is required a multiplication in Paillier encryption with distributed decryption is at least three times as costly as a multiplication in ElGamal encryption.

– If the modified ElGamal encryption is employed, the following drawbacks in efficiency exist.
  - Multiple encryption
    Usually, a voter has to perform an encryption for each candidate and prove each of his encryptions contains a valid message.
  - Inefficient DL search
    As stated before, a search for logarithm is needed in the decryption function. Even though the (currently known) most efficient solution for DL in a certain interval — Pollard's Lambda Method — is employed, $0.5 \log_2 n$ exponentiations, $O(n^{0.5})$ multiplications and $O(0.5 \log_2 n)$ storage are needed where n is the number of voters. As the number of voters is often large in voting applications, this is a high cost. To make the search more efficient, the votes may be divided into multiple groups and a separate tallying is performed in each group. However, this division increases the number of decryptions as a separate decryption is needed for every candidate in each group.

In [24, 25], the modified ElGamal encryption and its additive homomorphism are exploited in a very special way. Only one encryption is needed in a vote, which is composed of several sections, each corresponding to one candidate. So only one decryption is needed to decrypt the product of all the encrypted votes. Although the numbers of encryptions and decryptions are reduced, they are not the main computational burden in the voting scheme. The main computational burden of the voting scheme increases as the computational cost for vote validity proof increases and the cost of the DL search increases to $O(mn^{m-1})$ multiplications and $O(nm)$ full length (e.g. 1024 bits) storage space where m is the number of candidates. As the number of voters is often large in voting applications, the cost for the search is intolerable. So the special additive homomorphic tallying in [24, 25] actually deteriorates efficiency although it was supposed to improve efficiency.

In comparison, as will be illustrated in Section 3, multiplicative homomorphic voting employs efficient distributed key generation, requires only one encryption per vote and needs no DL search, while it achieves vote privacy no weaker than that of additive homomorphic voting.

3 The Multiplicative Homomorphic Voting Scheme

A multiplicative homomorphic voting scheme exploits multiplicative homomorphism of the encryption algorithm used for vote encryption to tally efficiently without revealing any vote. Each voter only needs to encrypt with a multiplicative homomorphic encryption algorithm one integer as his vote. An encryption algorithm is multiplicative homomorphic if $E(m_1m_2) = E(m_1)E(m_2)$ where $E()$ stands for the encryption function and m_1,m_2 are two random messages. A typical multiplicative homomorphic encryption algorithm is ElGamal encryption, which is employed in this paper. The product of the encrypted votes are then decrypted, so that the product of the votes is obtained if their product is not over the multiplicative modulus (certain mechanism is used to guarantee this assumption). Then the product is factorized to recover the votes. A voting protocol to elect one winner from m candidates is as follows.

1. Preparation phase
   Suppose there are m candidates $C_1, C_2, \ldots, C_m$. ElGamal encryption modulo p is employed for vote encryption where $p = 2q+1$ and p, q are large primes. Several talliers cooperate to generate and threshold share the private key while the public key is published using the distributed key generation function in [16]. A set $Q = \{q_1, q_2 \ldots, q_m\}$ is chosen to represent the candidates as follows.
   (a) Two sets $Q_1 = \{1\}$ and $Q_2 = \Phi$ are initialised. Two integers s_1 and s_2 representing the sizes of the two sets respectively are initialised as $s_1 = 1$ and $s_2 = 0$. Index s is initialised to be 1.
   (b) The s^{th} smallest prime p_s is tested.
   If $p_s^q = 1 \bmod p$,
   – p_s is a quadratic residue;
   – p_s is put into Q_1 and set $s_1 = s_1 + 1$.
   If $p_s^q \neq 1 \bmod p$,
   – p_s is not a quadratic residue;
   – p_s is put into Q_2 and set $s_2 = s_2 + 1$.
   (c) If $s_1 < m$ and $s_2 < m$, set $s = s + 1$ and go to Step (b). Otherwise, go to next step.
   (d) If $s_1 = m$, $Q = Q_1$; If $s_2 = m$, $Q = Q_2$.
   With this setting-up, the members in Q are either all quadratic residues or all quadratic non-residues, so their encryptions are indistinguishable[1].
   The talliers set up the ElGamal encryption:
   – they cooperatively generate the public key g and y in G, which is the subgroup in Z_p^* with order q using the distributed key generation techniques in [14, 30, 16], such that the private key $x = \log_g y$ are shared by them;
   – public key g and y are published.

[1] An alternative method to generate Q is to choose p and q such that the $m-1$ smallest primes are quadratic residues modulo p. Different large primes are tested as possible choices of p until a satisfying p is found. So, Q contains 1 and the $m-1$ smallest primes. However, it is not clear whether this method is feasible or efficient, especially when m is large.

2. Voting phase
   Each of the n voters $V_1, V_2, \ldots, V_n$ chooses a vote from Q. Voter V_i encrypts his vote v_i to $c_i = E(v_i) = (a_i, b_i) = (g^{r_i}, v_i y^{r_i})$ where r_i is randomly chosen from Z_q. V_i proves that an element in Q is encrypted in c_i without revealing his vote using the following honest-verifier ZK proof:

   $$\log_g a_i = \log_y(b_i/q_1) \ \vee \ \log_g a_i = \log_y(b_i/q_2) \ \vee \ldots \vee \ \log_g a_i = \log_y(b_i/q_m)$$

   This proof is based on the ZK proof of equality of logarithms [8] and the ZK proof of partial knowledge [10].
3. Tallying phase
   The talliers verify that every vote is valid. Then they randomly divide the encrypted votes $c_1, c_2, \ldots, c_n$ to groups of size k, so that $Max(Q)^k < p$ where $Max(Q)$ stands for the largest element in set Q. If $Max(Q)^n < p$, the division is not necessary and all the votes are in the same group. In each group the following multiplicative homomorphic tallying is performed.
   (a) Suppose $c'_1, c'_2, \ldots, c'_k$ are the encrypted votes in a group.
   (b) The talliers cooperate to calculate $v = D(\prod_{i=1}^{k} c'_i)$ where $D()$ denotes decryption.
   (c) v is factorized[2].
   - If $1 \notin Q$, $v = \prod_{j=1}^{m} p_j^{t_j}$ and the number of votes in this group for the j^{th} candidate is t_j for $j = 1, 2, \ldots, m$.
   - If $1 \in Q$, $v = \prod_{j=1}^{m-1} p_j^{t_j}$ and the number of votes in this group for the j^{th} candidate is t_{j-1} for $j = 2, 3, \ldots, m$ while the number of votes in this group for the first candidate is $k - \sum_{j=1}^{m-1} t_j$.

   The talliers sum up the results in all the groups to get the final result.

4 Analysis

The new voting scheme is analysed in this section to show that it is correct and efficient.

Theorem 1. *The multiplicative homomorphic tallying in each group with encrypted votes $c'_1, c'_2, \ldots, c'_k$ is correct.*

Proof: In the multiplicative homomorphic tallying in each group with encrypted votes $c'_1, c'_2, \ldots, c'_k$,

$$D(\prod_{i=1}^{k} c'_i) = v = \prod_{j=1}^{m-1} p_j^{t_j}$$

where $\prod_{j=1}^{m-1} p_j^{t_j}$ is a factorization of v.

As ElGamal encryption is multiplicative homomorphic,

$$D(\prod_{i=1}^{k} c'_i) = \prod_{i=1}^{k} D(c'_i) \bmod p$$

[2] This factorization is very efficient as each prime in Q is very small.

When the encrypted votes are divided into groups, it is guaranteed that $Max(Q)^k < p$. So $\prod_{i=1}^{k} D(c'_i) < p$ Therefore,

$$\prod_{i=1}^{k} D(c'_i) = D(\prod_{i=1}^{k} c'_i) = \prod_{j=1}^{m-1} p_j^{t_j}$$

As $D(c'_i)$ for $i = 1, 2, \ldots, k$ are verified to be in Q in the voting phase, $\prod_{i=1}^{k} D(c'_i)$ is also a factorization of v.

As there is a unique factorization for any integer, $\prod_{i=1}^{k} D(c'_i)$ and $\prod_{j=1}^{m-1} p_j^{t_j}$ are the same factorization. Namely, each prime factor in $\prod_{i=1}^{k} D(c'_i)$ is also a prime factor in $\prod_{j=1}^{m-1} p_j^{t_j}$ and each prime factor in $\prod_{j=1}^{m-1} p_j^{t_j}$ is also a prime factor in $\prod_{i=1}^{k} D(c'_i)$.

Therefore, all the non-one votes encrypted in $c'_1, c'_2, \ldots, c'_k$ and only these votes are prime factors in $\prod_{j=1}^{m-1} p_j^{t_j}$. That means every non-one vote is correctly recovered.

As the number of vote in each group is a constant k, the number of "1" votes is also correctly recovered if there are any. □

Theorem 2. *Multiplicative homomorphic tallying does not reveal any vote.*

Sketch of proof:

- Semantically secure encryption
  The usage of ElGamal encryption in this paper is semantically secure due to the choice of message space Q. (Either all members are quadratic residues or no member is quadratic residue where $p = 2q + 1$) So, without the private key to decrypt the votes, it is difficult to get any information about any vote.
- Private key (decryption) security
  As the private key is protected by a threshold key sharing mechanism, no single vote is decrypted if a threshold trust on the talliers is assumed.
- Unlinkability (No bidder can be linked to his bid.)
  As a result, the only message decrypted from the encrypted votes is the product of votes in each group, which links no vote to the corresponding voter. The revealed information tells no more than that a voter in every group may have submitted any vote in the group.
- The group size is large enough for strong vote privacy.
  As homomorphic tallying is only applied to elections with a small number of candidates, m and $Max(Q)$ are small[3]. As p is large (e.g. with a length of 1024 bits), $\lceil \log_{Max(Q)} p \rceil$, the size of a group, is large compared to m where $\lceil x \rceil$ denotes the smallest integer no smaller than a real number x. For example, when $m = 2$ and $|p| = 1024$ where $||$ stands for bit length, we get $Q = \{1, 2\}$ (for simplicity, assuming 2 is a quadratic residue), $Max(Q) = 2$

[3] $Max(Q)$ is no larger than the $(2m-1)^{th}$ smallest prime, which is no more than a small multiple of m when m is small.

and the group size is larger than 1024. When there are only two candidates and more than 1024 votes are mixed together in each group, strong vote privacy is achieved.

□

Every operation in the voting scheme can be publicly verified by anyone. (Note that public proofs of vote validity and correctness of decryption are provided by the voters and talliers respectively.) The computational cost of additive homomorphic voting employing Paillier encryption and that of the proposed multiplicative homomorphic voting are listed in Table 1. As stated in Section 2, the DL search in the decryption of the modified ElGamal encryption in [18, 22, 24, 25] is too inefficient[4]. So the modified ElGamal encryption is not considered as a good choice in additive homomorphic voting. As only small primes are employed to stand for the votes, the computational cost of the final factorization in multiplicative homomorphic voting is negligible compared to full length exponentiation. To make a precise comparison of the efficiency of the two kinds of homomorphic voting, it is supposed the same strength of encryption security is required in both kinds of voting, so N in Paillier encryption employed in additive homomorphic voting and p in ElGamal encryption employed in multiplicative homomorphic voting have the same length. An exponentiation in multiplicative homomorphic voting (employing ElGamal encryption) is called a standard exponentiation while an exponentiation in additive homomorphic voting (employing Paillier encryption with distributed decryption) is accounted as three standard exponentiations. The number of standard exponentiations is accounted in every operation in Table 1. This table clearly illustrates that multiplicative homomorphic voting is always more efficient than additive homomorphic voting in key generation, vote encryption and vote validity check. When the number of voters is not too large, multiplicative homomorphic voting is also more efficient than additive homomorphic voting in tallying. For example, when $m = 2$, $|p| = 1024$ and $n = 1024$, the needed number of standard exponentiation in tallying in additive homomorphic voting is 12 or 6, while the needed number of standard exponentiation in tallying in multiplicative homomorphic voting is 3. Even if multiplicative homomorphic tallying is less efficient than additive homomorphic tallying when the number of voters is large, it has a trivial influence on the total cost of the voting scheme as will be shown in Table 2.

A more comprehensive efficiency comparison is presented in Table 2, where the efficiency of MV (mix voting), AHV (additive homomorphic voting) and the proposed MHV (multiplicative homomorphic voting) are compared. In this comparison, [17] (one of the most efficient mix voting) is taken as an example of mix voting where ElGamal encryption is employed. It is assumed that the additive homomorphic voting (no existing example is referred to) employs distributed

[4] Although some computation in the Pollard's Lambda Method can be pre-computed, precomputation can be employed in most voting schemes. For example, the exponentiation computation in vote encryption and all the computation in the proof of vote validity (if necessary) can be precomputed in mix voting, Paillier-based additive homomorphic voting and multiplicative homomorphic voting.

Table 1. Computational cost of the two kinds of homomorphic voting

	Additive homomorphic voting	Multiplicative homomorphic voting
Distributed key generation	highly inefficient	efficient
Encryption per vote	$6m$	2
Vote validity proof per vote	$12m + 6$	$4m - 2$
Vote validity verification per vote	$12m + 6$	$4m$
Tallying computation per tallier	$9m$ or[a] $9(m-1)$	$3\lceil n \log_p Max(Q) \rceil$

[a] It is often assumed that a decryption is necessary for every candidate. However, when n, the total number of voters is known and each vote has been verified to be valid, $m-1$ decryptions are enough. The talliers randomly choose $m-1$ candidates and decrypt the sum of votes for each of them. The vote of the remaining candidate is n minus the sum of the votes for the $m-1$ chosen candidates. We call this economical tallying.

Paillier encryption and performs every necessary operation listed in Table 1. For simplicity, voters' signature on the votes are omitted, so voters' signature generation and talliers' signature verification are not taken into account. In Table 2, t is the number of talliers. The number of standard exponentiation is accounted in computational cost and the number of transported bits is accounted in communicational cost. An example is given in Table 2, where $t = 5$, $m = 2$, $|p| = 1024$ and $n = 1000000$. For simplicity, it is assumed that 2 is a quadratic residue modulo p, so $Q = 1, 2$. In this example, it is shown that even when the number of voters is large, multiplicative homomorphic voting is still more efficient than mix voting and additive homomorphic voting. When the number of voters is not large and grouping is not necessary in tallying, efficiency advantage of multiplicative homomorphic voting is more obvious. So multiplicative homomorphic voting is the most efficient voting solution when the number of candidates is small.

5 Conclusion

The voting scheme in this paper employs a new tallying method: multiplicative homomorphic tallying. It achieves the highest efficiency when the number of candidates is small and guarantees strong vote privacy and public verifiability.

Table 2. Efficiency comparison

	Key generation	Voter's computation	Tallier's computation	Communication
MV	efficient	8	$18n$ $= 18000000$	$1024(6n + 18tn)$ $=98304000000$
AHV[a]	highly inefficient	$18m + 6$ $= 42$	$(12m + 6)n$ $+9(m - 1)$ $= 30000009$	$2048(6t(m - 1)+$ $(10m + 4)n)$ $=49152061440$
MHV	efficient	$4m$ $= 8$	$4mn + 3\lceil n$ $\log_p Max(Q)\rceil$ $= 8003000$	$1024(2n(m + 1)+$ $3\lceil n \log_p Max(Q)\rceil)$ $=6147072000$

[a] It is assumed economical tallying in Table 1 is employed.

References

1. Masayuki Abe and Hideki Imai. Flaws in some robust optimistic mix-nets. In *Advances in Cryptology—ACISP 03*, pages 39–50, 2003.
2. R. Aditya, C. Boyd, E. P. Dawson, and K. Viswanathan. Secure e-voting for preferential elections. In *Proceedings of EGOV 03 Conference*, pages 246–249, Berlin, 2003. Springer-Verlag. Lecture Notes in Computer Science Volume 2738.
3. James M. Adler, Wei Dai, Richard L. Green, and C. Andrew Neff. Computational details of the votehere homomorphic election system. Technical report, VoteHere Inc, 2000. Available from http://www.votehere.net/technicaldocs/hom.pdf, last accessed 22 June 2002.
4. Olivier Baudron, Pierre-Alain Fouque, David Pointcheval, Jacques Stern, and Guillaume Poupard. Practical multi-candidate election system. In *Twentieth Annual ACM Symposium on Principles of Distributed Computing*, pages 274–283, 2001.
5. Josh Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 544–553, 1994.
6. Josh Daniel Cohen Benaloh. *Verifiable Secret-Ballot Elections*. PhD thesis, Faculty of Graduate School, Yale University, 1996.
7. Dan Boneh and Philippe Golle. Almost entirely correct mixing with applications to voting. In *9th ACM Conference on Computer and Communications Security—CCS 02*, pages 68–77, 2002.
8. D. Chaum and T. P. Pedersen. Wallet databases with observers. In *CRYPTO '92*, pages 89–105, Berlin, 1992. Springer-Verlag. Lecture Notes in Computer Science Volume 740.
9. David Chaum. Secret-ballot receipts: True voter-verifiable elections. *IEEE Security and Privacy*, 2(1):38–47, January/February 2004.
10. R. Cramer, I. B. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, pages 174–187, Berlin, 1994. Springer-Verlag. Lecture Notes in Computer Science Volume 839.
11. Ronald Cramer, Matthew Franklin, Berry Schoenmakers, and Moti Yung. Multi-authority secret-ballot elections with linear work. In *Advances in Cryptology—EUROCRYPT 96*, pages 72–83, 1996.

12. Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A secure and optimally efficient multi-authority election scheme. In *Advances in Cryptology—EUROCRYPT 97*, pages 103–118, 1997.
13. Ivan Damgaård and Mats Jurik. A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In *Public Key Cryptography—PKC 01*, pages 119–136, 2001.
14. P Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science*, pages 427–437, 1987.
15. Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In *Advances in Cryptology—AUSCRYPT 92*, pages 244–251, 1992.
16. R Gennaro, S Jarecki, H Krawczyk, and T Rabin. Secure distributed key generation for discrete-log based cryptosystems. In *EUROCRYPT '99*, pages 123–139, Berlin, 1999. Springer-Verlag. Lecture Notes in Computer Science Volume 1592.
17. Philippe Golle, Sheng Zhong, Dan Boneh, Markus Jakobsson, and Ari Juels. Optimistic mixing for exit-polls. In *Advances in Cryptology—ASIACRYPT 02*, pages 451–465, 2002.
18. Alejandro Hevia and Marcos Kiwi. Electronic jury voting protocols. 2000. http://eprint.iacr.org/2000/035/.
19. Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. In *Advances in Cryptology—EUROCRYPT 00*, pages 539–556, 2000.
20. Markus Jakobsson, Ari Juels, and Ronald L. Rivest. Making mix nets robust for electronic voting by randomized partial checking. In *11th USENIX Security Symposium*, pages 339–353, 2002.
21. Jonathan Katz, Steven Myers, and Rafail Ostrovsky. Cryptographic counters and applications to electronic voting. In *Advances in Cryptology—EUROCRYPT 01*, pages 78–92, 2001.
22. Aggelos Kiayias and Moti Yung. Self-tallying elections and perfect ballot secrecy. In *Public Key Cryptography, 5th International Workshop—PKC 02*, pages 141–158, 2002.
23. Byoungcheon Lee, Colin Boyd, Ed Dawson, Kwangjo Kim, Jeongmo Yang, and Seungjae Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *to appear in Information Security and Cryptology, ICISC 2003*, 2003.
24. Byoungcheon Lee and Kwangjo Kim. Receipt-free electronic voting through collaboration of voter and honest verifier. In *JW-ISC 2000*, pages 101–108, 2000.
25. Byoungcheon Lee and Kwangjo Kim. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In *Information Security and Cryptology, ICISC 2002*, pages 389–406, 2002.
26. C. Andrew Neff. Conducting a universally verifiable electronic election using homomorphic encryption. White paper, VoteHere Inc, 2000.
27. C. Andrew Neff. Verifiable, secret shuffles of elgamal encrypted data for secure multi-authority elections. In *8th ACM Conference on Computer and Communications Security—CCS 01*, pages 116–125, 2001.
28. Tatsuaki Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Proc. Security Protocols, 5th International Workshop 1997*, pages 25–35, 1997.
29. P Paillier. Public key cryptosystem based on composite degree residuosity classes. In *EUROCRYPT '99*, pages 223–238, Berlin, 1999. Springer-Verlag. Lecture Notes in Computer Science Volume 1592.
30. Torben P. Pedersen. A threshold cryptosystem without a trusted party. In *EUROCRYPT '91*, pages 522–526, Berlin, 1991. Springer-Verlag. Lecture Notes in Computer Science Volume 547.

31. Andreu Riera and Joan Borrell. Practical approach to anonymity in large scale electronic voting schemes. In *Network and Distributed System Security Symposium—NDSS 99*, pages 69–82, 1999.
32. Andreu Riera, Josep Rifà, and Joan Borrell. Efficient construction of vote-tags to allow open objection to the tally in electronic elections. *Information Processing Letters*, 75(5):211–215, October 2000.
33. Kazue Sako and Joe Kilian. Secure voting using partially compatible homomorphisms. In *Advances in Cryptology—CRYPTO 94*, pages 411–424, 1994.
34. Kazue Sako and Joe Kilian. Receipt-free mix-type voting scheme: A practical solution to the implementation of a voting booth. In *Advances in Cryptology—EUROCRYPT 95*, pages 393–403, 1995.
35. Berry Schoenmakers. Fully auditable electronic secret-ballot elections. *XOOTIC Magazine*, July 2000.
36. Douglas Wikström. How to break, fix and optimize "optimistic mix for exit-polls". Technical report, Swedish Institute of Computer Science, 2002. Available from http://www.sics.se/libindex.htlm, last accessed 08 October 2003.

Chosen Ciphertext Attack on a New Class of Self-Synchronizing Stream Ciphers*

Bin Zhang[1,2], Hongjun Wu[1], Dengguo Feng[2], and Feng Bao[1]

[1] Institute for Infocomm Research, Singapore
[2] State Key Laboratory of Information Security,
Graduate School of the Chinese Academy of Sciences,
Beijing 100039, P.R. China
zhangbin@mails.gscas.ac.cn
{hongjun, baofeng}@i2r.a-star.edu.sg

Abstract. At Indocrypt'2002, Arnault et al. proposed a new class of self-synchronizing stream ciphers combining LFSR and FCSR architectures. It was claimed to be resistant to known attacks. In this paper, we show that such a self-synchronizing stream cipher is extremely vulnerable to chosen ciphertext attack. We can restore the secret keys easily from one chosen ciphertext with little computation. For the parameters given in the original design, it takes less than one second to restore the secret keys on a Pentium 4 processor.

Keywords: Stream cipher, Self-synchronizing, 2-adic expansion, Feedback shift register.

1 Introduction

Stream ciphers are an important class of encryption algorithms in practice. In general, they are classified into two kinds: synchronous stream ciphers and self-synchronous stream ciphers [5]. In a self-synchronizing stream cipher, each plaintext bit affects the entire following ciphertext through some mechanism, which makes it more likely to be resistant against attacks based on plaintext statistical properties. Since several ciphertext bits may be incorrectly decrypted when a bit modification occurs in the ciphertext, such a mechanism provides additional security against active attacks.

In [1], a new class of self-synchronous stream ciphers was proposed which exploits the concatenation of LFSR and FCSR. The main idea behind such a design is to confuse the $GF(2)$ linearity with the 2-adic linearity so that neither of the synthesis algorithms (Berlekamp-Massey type algorithms) can work in this case. However, as we will show in this paper, such a simple design is extremely weak under chosen ciphertext attack. By choosing one ciphertext, we can recover

* Supported by National Natural Science Foundation of China (Grant No. 60273027), National Key Foundation Research 973 project (Grant No. G1999035802) and National Science Fund for Distinguished Young Scholars (Grant No. 60025205).

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 73–83, 2004.

the secret keys with little computation. Assume both LFSR and FCSR are of length 89, as suggested by the authors [1], we can recover both the structures in 1 second on a Pentium 4 processor.

This paper is organized as follows. In Section 2, we will give an introduction to the self-synchronizing stream cipher together with some backgrounds. Our attack on this cipher is given in Section 3 and detailed experimental results are also included in this section. Finally, some conclusions are given in Section 4.

2 The Self-Synchronizing Stream Cipher

In this section, we will first review some backgrounds including 2-adic arithmetic and the Galois representations of LFSR and FCSR. Then a detailed description of the self-synchronizing stream cipher is presented.

2.1 2-Adic Arithmetic, Galois Representations of LFSR and FCSR

A 2-adic integer is a formal power series $s = \sum_{i=0}^{\infty} s_i 2^i$ with $s_i \in \{0, 1\}$. We denote the set of 2-adic integers by $\mathbb{Z}_2$. The addition and multiplication in $\mathbb{Z}_2$ is done according to $2^i + 2^i = 2^{i+1}$ for all $i \geq 0$, i.e. taking the carry to the higher order term. Thus the addition inverse of 1 is $\sum_{i=0}^{\infty} 2^i = -1$ and if a 2-adic integer $s = 2^r + \sum_{i=r+1}^{\infty} s_i 2^i$, its addition inverse is $-s = 2^r + \sum_{i=r+1}^{\infty} (1 - s_i) 2^i$.

A feedback with carry shift register (FCSR) is a device for the fast generation of pseudorandom sequence with good statistical properties and large period. Like LFSR, FCSR also has two architectures: Fibonacci structure and Galois structure [3]. The Galois architecture is more efficient due to the parallel computation of feedbacks. As in [1], we only consider the Galois structure in this paper.

Lemma 1 characterizes the eventually periodic binary sequences in terms of 2-adic integers.

Lemma 1. *[3] Let $S_2 = \sum_{i=0}^{\infty} s_i 2^i$ be the 2-adic integer corresponding to a binary sequence $S = \{s_i\}_{i \geq 0}$. S is eventually periodic if and only if there exists two integers p and q in $\mathbb{Z}$ such that $S_2 = p/q$ with q odd, see Figure 1. Further, S is strictly periodic if and only if $pq \leq 0$ and $|p| \leq |q|$.*

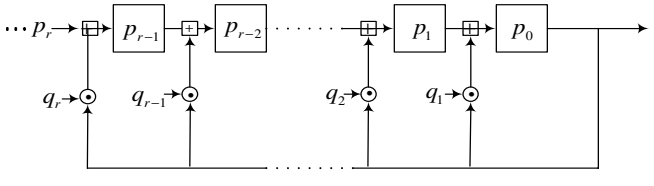

Fig. 1. Galois representation of a FCSR

The 2-adic division of p/q is fulfilled by a FCSR using Galois architecture. Without loss of generality, we always assume $p = \sum_{i=0}^{r} p_i 2^i + p_{r+1} 2^{r+1} + \cdots \geq 0$ and $q = 1 - \sum_{i=1}^{r} q_i 2^i < 0$ with p_i and $q_i \in \{0, 1\}$. In Figure 1, $\boxplus$ denotes the

addition with carry, i.e. the output of $a \boxplus b$ is $a \oplus b \oplus c_{n-1}$ and the carry is $c_n = ab \oplus ac_{n-1} \oplus bc_{n-1}$. $\odot$ is the binary multiplication. The period of S is the smallest integer t such that $2^t \equiv 1 \pmod{q}$.

Similarly, the Galois representation of a LFSR is shown in Figure 2, where the input of the circuit is $S(x) = \sum_{i=0}^{\infty} s_i x^i$ and the output is $S'(x) = S(x)/Q(x)$, with $Q(x) = 1 + \sum_{i=1}^{r} q_i x^i$ and $\oplus$ being Xor.

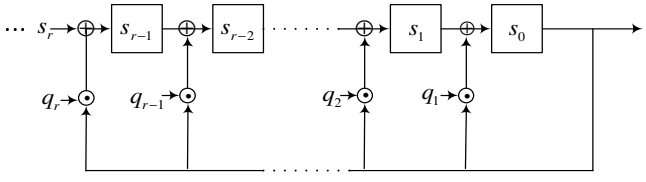

Fig. 2. Galois representation of a LFSR

2.2 Description of the Self-Synchronizing Stream Cipher

The structure of this cipher is a concatenation of one LFSR and one FCSR, as shown in Figure 3. An irreducible primitive polynomial $Q(x)$ of prime degree k is

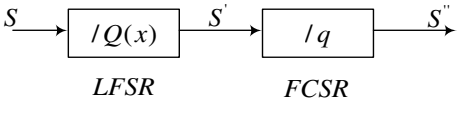

Fig. 3. The self-synchronizing stream cipher

used as the feedback polynomial of the LFSR. A negative prime q for the FCSR divisor box is of size k satisfying $|q| = 2u+1$ with u a prime congruent to 1 modulo 4 and $\gcd(|q|-1, 2^k-1) = 1$, where $|\cdot|$ denotes the absolute value. For practical applications, the authors of [1] suggest $k = 89$. Initialize LFSR and FCSR randomly and denote the message to be encrypted by S, the encryption scheme is:

1. Compute $S'(x) = S(x)/Q(x)$ by the LFSR divisor-box.
2. Convert $S'(x)$ into the 2-adic integer $S'(2)$.
3. Compute the ciphertext $S'' = S'(2)/q$ by the FCSR divisor-box.

Upon decrypting, without loss of generality, initialize all the LFSR and FCSR cells (including the carries) to be zero. The corresponding decryption scheme is:

1. Compute $S'(2) = qS''(2)$.
2. Convert $S'(2)$ into the formal power series $S'(x)$.
3. Compute the plaintext S by $S(x) = Q(x)S'(x)$.

The decryption circuits are only generally discussed in [1], no concrete circuit is given in that paper. It is claimed in [1] that this cipher is fast and secure against known attacks. However, as we will show below, this self-synchronizing stream cipher is far away from security.

3 Our Attack

In this section, we will show that this self-synchronizing stream cipher is very vulnerable to chosen ciphertext attack. Subsection 3.1 gives the circuits fulfilling multiplication by q and $Q(x)$, respectively. Our attack is given in subsection 3.2. The experimental results are given in subsection 3.3 in detail.

3.1 The Multiplication Circuits

Without loss of generality, we propose the following two circuits to fulfill the multiplication by q and $Q(x)$, respectively. The only purpose of this section is to show that the proposed stream cipher can actually decrypt the ciphertext.

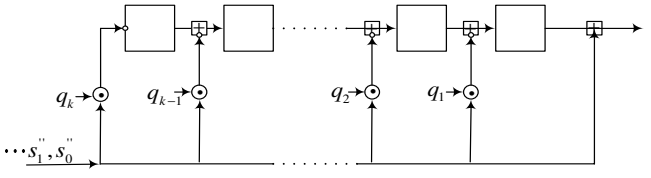

Fig. 4. Multiplication circuit with q being the multiplier

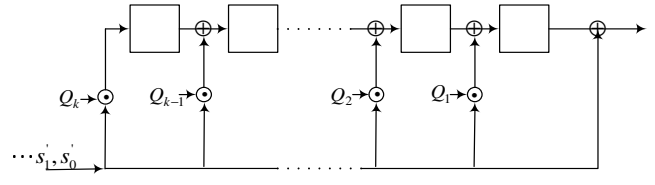

Fig. 5. Multiplication circuit with $Q(x)$ being the multiplier

In Figures 4 and 5, $q = 1 - \sum_{i=1}^{k} q_i 2^i$ and $Q(x) = 1 + \sum_{i=1}^{k} Q_i x^i$ are the secret parameters used in the cipher. Upon decrypting, let all the cells in Figures 4 and 5 (including the carries) be zero. From (1), the inputs corresponding to the k cells in Figure 4 are the addition inverses of the ciphertext, while the input corresponding to rightmost $\boxplus$ is the ciphertext itself. We denote the inverse operation by a hollow circle in Figure 4.

$$\begin{aligned}
&(s_0'' + s_1'' \cdot 2^1 + s_2'' \cdot 2^2 + \cdots)(1 - q_1 \cdot 2^1 - q_2 \cdot 2^2 - \cdots - q_k \cdot 2^k) \\
&= -(s_0'' + s_1'' \cdot 2^1 + s_2'' \cdot 2^2 + \cdots)(q_1 \cdot 2^1 + q_2 \cdot 2^2 + \cdots + q_k \cdot 2^k - 1) \\
&= -(s_0'' + s_1'' \cdot 2^1 + s_2'' \cdot 2^2 + \cdots)(q_1 \cdot 2^1 + q_2 \cdot 2^2 + \cdots + q_k \cdot 2^k) + (s_0'' + s_1'' \cdot 2^1 \\
&\quad + s_2'' \cdot 2^2 + \cdots).
\end{aligned} \tag{1}$$

3.2 A Chosen Ciphertext Attack

The basic idea of our attack is that if we choose a special ciphertext fed into the decryption circuits such that the corresponding decrypted message (including a preamble) is of finite length, then we can retrieve the secret keys q and $Q(x)$ by simple factoring the polynomial corresponding to the decrypted message over $GF(2)$ [2, 6].

Since the secret q is a negative prime, the decrypted message is of finite length if the ciphertext fed into the decryption circuits is a binary string with the following form:

$$(\underbrace{*, *, \cdots, *}_{A}, \underbrace{1, 1, \cdots, 1, 1}_{B}), \tag{2}$$

where A is a randomly-chosen binary prefix of certain length and B is a all-1 string of certain length. We use a randomly-chosen string A to disguise the subsequent all-1 string. (3) confirms the validity of the above chosen ciphertext.

$$\begin{aligned} & (s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l + 1 \cdot 2^{l+1} + 1 \cdot 2^{l+2} + \cdots)q \\ = & (s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l)q \\ & + (1 \cdot 2^{l+1} + 1 \cdot 2^{l+2} + \cdots)q \\ = & (s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l)q + \frac{2^{l+1}}{1-2}q \\ = & (s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l)q + 2^{l+1} \cdot (-q) \\ & \rightarrow D(x), \end{aligned} \tag{3}$$

where $A = (s_0'', s_1'', \cdots, s_l'')$ is of length l. From (3), $(s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l)q + 2^{l+1} \cdot (-q)$ corresponds to a polynomial $D(x)$ of finite degree. Therefore, the polynomial corresponding to the decrypted message is $D(x)Q(x) \in GF(2)[x]$. According to [6], the following lemma holds.

Lemma 2. *[6] A univariate polynomial of degree n over the finite field $GF(p^k)$, where p is a small, fixed prime, can be factored with a deterministic algorithm whose running time is $O((nk)^2)$.*

From our experimental results, the degree of $D(x)Q(x)$ is less than 300. Hence, the complexity of factoring $D(x)Q(x)$ over $GF(2)$ is only $O(2^{16})$.

Upon factoring the decrypted message, we get both $Q(x)$ and $D(x)$. Let $D(x) = d_0 + d_1x^1 + d_2x^2 + \cdots + d_hx^h$. Keeping in mind that $D(x)$ is the polynomial representation of $(s_0'' + s_1'' \cdot 2^1 + \cdots + s_l'' \cdot 2^l)q + 2^{l+1} \cdot (-q)$, $d_0 + d_1 \cdot 2^1 + d_2 \cdot 2^2 + \cdots + d_h \cdot 2^h$ has an integer factor $(-q)$. Therefore, factoring the integer $d_0 + d_1 \cdot 2^1 + d_2 \cdot 2^2 + \cdots + d_h \cdot 2^h$ retrieves the secret q. Since $2^{89} \leq (-q) \leq 2^{90}$, $(-q)$ is an integer with at most 28 decimal digits which can be recovered easily by factoring $d_0 + d_1 \cdot 2^1 + d_2 \cdot 2^2 + \cdots + d_h \cdot 2^h$ using the number field sieve algorithm [4].

A full description of our attack is as follows.

1. Choose a string as shown in (2) and feed it into the decryption circuits.

2. Convert the decrypted message into polynomial form and factor it to get $Q(x)$ and $D(x)$.
3. Transform $D(x)$ into the integer form and factor the integer to recover q.

The complexity of our attack is very low for the parameter $k = 89$. See Subsection 3.3.

3.3 Experimental Results

We have implemented the above attack on a Pentium 4 processor, see Appendix A for the C source codes. The parameters of this self-synchronizing stream cipher are chosen as follows.

$$Q(x) = x^{89} + x^6 + x^5 + x^3 + 1, \;\; q = -618970052618499608724417827, \quad (4)$$

where $Q(x)$ is a primitive polynomial of degree 89 and q is a negative prime satisfying the following three conditions:

1. $2^{89} \leq (-q) \leq 2^{90}$.
2. $\gcd(618970052618499608724417826, 2^{89} - 1) = 1$.
3. $|q| = 2u + 1$ with $u = 309485026309249804362208913$ is a prime congruent to 1 modulo 4.

These three conditions are used to verify the candidate keys obtained from the attack. We choose a 600-bit ciphertext as follow.

$$A = (10101110101010001101001100110010011100 01) \,||\, B = (11 \cdots 11), \quad (5)$$

where A is a randomly-chosen string of length 40 and B is a all-1 string of length 560. Feed the above chosen ciphertext into the decryption circuits and get the result. The polynomial corresponding to the decrypted message is $x^{216} + x^{215} + x^{214} + x^{210} + x^{209} + x^{207} + x^{206} + x^{203} + x^{202} + x^{199} + x^{198} + x^{196} + x^{194} + x^{192} + x^{190} + x^{183} + x^{181} + x^{179} + x^{178} + x^{176} + x^{174} + x^{173} + x^{167} + x^{166} + x^{165} + x^{163} + x^{159} + x^{155} + x^{153} + x^{149} + x^{148} + x^{147} + x^{145} + x^{144} + x^{143} + x^{140} + x^{139} + x^{136} + x^{133} + x^{132} + x^{131} + x^{128} + x^{126} + x^{123} + x^{122} + x^{120} + x^{118} + x^{113} + x^{111} + x^{110} + x^{105} + x^{104} + x^{103} + x^{101} + x^{100} + x^{99} + x^{98} + x^{96} + x^{95} + x^{92} + x^{91} + x^{89} + x^{88} + x^{85} + x^{78} + x^{75} + x^{74} + x^{73} + x^{72} + x^{71} + x^{67} + x^{64} + x^{61} + x^{60} + x^{59} + x^{56} + x^{52} + x^{51} + x^{49} + x^{42} + x^{40} + x^{35} + x^{34} + x^{32} + x^{29} + x^6 + x^5 + x^3 + 1$. It takes 0.203 seconds to factor the above polynomial using Mathematica. The result is $(1 + x + x^4 + x^5 + x^6)(1 + x^7 + x^9 + x^{11} + x^{13} + x^{14} + x^{15})(1 + x + x^2 + x^5 + x^8 + x^9 + x^{11} + x^{16} + x^{17} + x^{19} + x^{20} + x^{21} + x^{24} + x^{25} + x^{26})(1 + x^3 + x^8 + x^9 + x^{10} + x^{11} + x^{15} + x^{17} + x^{18} + x^{22} + x^{24} + x^{25} + x^{29} + x^{30} + x^{31} + x^{32} + x^{33})(1 + x^4 + x^5 + x^6 + x^8 + x^9 + x^{13} + x^{15} + x^{19} + x^{20} + x^{22} + x^{23} + x^{24} + x^{26} + x^{27} + x^{30} + x^{31} + x^{33} + x^{34} + x^{35} + x^{36} + x^{37} + x^{40} + x^{42} + x^{43} + x^{44} + x^{45} + x^{46} + x^{47})(1 + x^3 + x^5 + x^6 + x^{89})$. Expand the factors other than $x^{89} + x^6 + x^5 + x^3 + 1$ and let $x = 2$. The result is 302266531961499475785005717448795619329. By Mathematica, it takes 0.219 seconds to factor this integer. The result is $3^2 \cdot 23 \cdot 8839 \cdot 266899 \cdot 618970052618499608724417827$. Therefore, the total time required to restore $Q(x)$ and q is about 0.422 seconds.

Some Remarks. A pseudorandom generator with a similar structure was also proposed in [1]. Since our attack only work in the chosen ciphertext scenario, the security of that generator is not influenced by our attack.

4 Conclusion

In this paper, we showed that the proposed self-synchronizing stream cipher is extremely weak against chosen ciphertext attack. We can restore the secret keys easily from a 600-bit chosen ciphertext in 1 second on a Pentium 4 processor. We suggest that this cipher should not be used in practice.

References

1. F. Arnault, T. P. Berger, A. Necer, "A New Class of Stream Ciphers Combining LFSR and FCSR Architectures", *Progress in Cryptology-INDOCRYPT'2002*, LNCS vol. 2551, Springer-Verlag,(2002), pp. 22-33.
2. Berlekamp, E. R., "Factoring polynomials over finite fields", *Bell Systems Tech. J.*, 46, 1967, pp. 1853-1859.
3. M. Goresky, A. M. Klapper, "Fibonacci and Galois Representations of Feedback-With-Carry Shift Registers", *IEEE Transactions on Information Theory*, vol. 48, No. 11, 2002, pp. 2826-2836.
4. A. K. Lenstra, H. W. Lenstra, Jr., M. S. Manasse, J. M. Pollard, "The Number Field Sieve", *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing*, Baltimore, Maryland, May 1990, pp. 14-16.
5. A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press,1997.
6. V. Shoup, "A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic", *Proc. 1991 International Symposium on Symbolic and Algebraic Computation*, 1991, pp. 14-21.

A A Non-optimized C Implementation of Our Attack

```
#include "stdio.h"
#include "math.h"
#define SIZE 8000

unsigned char LFSR[89],FCSR[89],carry[89],cipher[SIZE],
re[SIZE],fe[SIZE],inter[SIZE],plain[SIZE],deinter[SIZE];

void main()
{
 unsigned char S[256],z[256],G[5]={179,43,228,11,194};
 unsigned char RF[90],RL[90],i1,j1,L[90],F[90];
 unsigned int i,j,n,k1;
```

```
//use RC4 as the random source
for(k1=0;k1<256;k1++)S[k1]=k1;
 for(j1=0,k1=0;k1<=255;k1++)
 {
     j1=j1+S[k1]+G[k1%5];
     i1=S[k1];S[k1]=S[j1];S[j1]=i1;
 }

 for(i1=0,j1=0,k1=0;k1<256;k1++)
 {
     i1++;j1=j1+S[i1];n=S[i1];S[i1]=S[j1];S[j1]=n;
     z[k1]=S[(unsigned char)(S[i1]+S[j1])];
 }

 //initialization
 for(i=0;i<SIZE;i++)
     inter[i]=cipher[i]=plain[i]=deinter[i]=re[i]=fe[i]=0;
 for(i=0;i<89;i++)
     LFSR[i]=FCSR[i]=carry[i]=0;
 for(i=0;i<300;i++)
     for(j=0;j<8;j++)
     {
         deinter[8*i+(j%8)]=(z[i]&(1<<j))>>j;
     }

 //printf("The message is: \n");
 //for(i=0;i<300;i++)printf("%x ",deinter[i]);printf("\n");

 // for simplicity, we just initial the LFSR
 // and FCSR as follows.
 LFSR[0]=1;
 LFSR[1]=1;
 LFSR[45]=1;
 //LFSR[3]=1;
 LFSR[88]=1;
 //FCSR[1]=1;
 FCSR[0]=1;
 FCSR[29]=1;
 //FCSR[3]=1;
 FCSR[88]=1;

 for(i=0;i<90;i++)
     L[i]=F[i]=0;
 for(i=0;i<90;i++)
     RL[i]=RF[i]=0;
```

```
L[0]=L[3]=L[5]=L[6]=L[89]=1;
F[0]=F[2]=F[5]=F[8]=F[10]=1;
F[11]=F[13]=F[15]=F[18]=F[19]=F[22]=1;
F[26]=F[28]=F[29]=F[42]=F[43]=F[44]=1;
F[47]=F[48]=F[53]=F[55]=F[56]=F[59]=1;
F[62]=F[63]=F[64]=F[89]=1;
for(i=0;i<600;i++)
{
    inter[i]=(LFSR[0] & 1);
    j1=deinter[i];
    for(j=0;j<90;j++)
        RL[j]=(L[j] & LFSR[0]);
    for(j=0;j<88;j++)
        LFSR[j]=RL[j+1]^LFSR[j+1];
    LFSR[88]=RL[89]^j1;
}


for(i=0;i<600;i++)
{
    j1=inter[i];
    cipher[i]=FCSR[0];
    for(j=0;j<90;j++)
        RF[j]=(F[j] & FCSR[0]);
    for(j=0;j<88;j++)
    {
      FCSR[j]=(FCSR[j+1]+RF[j+1]+carry[j])&1;
      carry[j]=
      (unsigned char)((FCSR[j+1]+RF[j+1]+carry[j])&(1<<1))>>1;
    }
    FCSR[88]=(RF[89]+j1+carry[88])&1;
    carry[88]=
    (unsigned char)((RF[89]+j1+carry[88])&(1<<1))>>1;
}

//printf("The ciphertext is: \n");
//for(i=0;i<300;i++)printf("%x ",cipher[i]);printf("\n");

// our attack
for(i=40;i<600;i++)
    cipher[i]=1;
printf("***\n");
for(i=0;i<40;i++)
    printf("%x ", cipher[i]);
printf("***\n");
```

```
for(i=0;i<600;i++)
    re[i]=cipher[i];

i=0;
while (cipher[i]==0) i=i+1;
for(n=i+1;n<600;n++)
    cipher[n]=cipher[n]^1;

for(i=0;i<90;i++)
    L[i]=F[i]=0;
for(i=0;i<89;i++)
    FCSR[i]=carry[i]=0;
for(i=0;i<90;i++)
    RL[i]=RF[i]=0;
L[0]=L[3]=L[5]=L[6]=L[89]=1;
F[0]=F[2]=F[5]=F[8]=F[10]=1;
F[11]=F[13]=F[15]=F[18]=F[19]=F[22]=1;
F[26]=F[28]=F[29]=F[42]=F[43]=F[44]=1;
F[47]=F[48]=F[53]=F[55]=F[56]=F[59]=1;
F[62]=F[63]=F[64]=F[89]=1;
for(i=0;i<600;i++)
{
    j1=cipher[i];
    i1=re[i];
    fe[i]=((FCSR[0]+i1+carry[0])&1);
    carry[0]=
    (unsigned char)((FCSR[0]+i1+carry[0])&(1<<1))>>1;
    for(j=1;j<90;j++)
        RF[j]=(j1 & F[j]);
    for(j=0;j<88;j++)
    {
      FCSR[j]=(RF[j+1]+FCSR[j+1]+carry[j+1])&1;
      carry[j+1]=
      (unsigned char)((RF[j+1]+FCSR[j+1]+carry[j+1])&(1<<1))>>1;
    }
    FCSR[88]=j1;
}

for(i=0;i<89;i++)
    LFSR[i]=0;
for(i=0;i<600;i++)
{
    j1=fe[i];
    plain[i]=((LFSR[0]^j1)&1);
    for(j=1;j<90;j++)
```

```
            RL[j]=(L[j] & j1);
        for(j=0;j<88;j++)
            LFSR[j]=(LFSR[j+1]^RL[j+1]);
        LFSR[88]=j1;
    }


    printf("The decrypted message is: \n");
    for(i=0;i<300;i++)
        printf("%x ",plain[i]);
    printf("\n");

}
```

After getting the result, we can use Mathematica to restore the secret keys.

Algebraic Attacks over $GF(q)$

Lynn Margaret Batten*

Deakin University, 221 Burwood Highway, Vic 3125, Australia
lmbatten@deakin.edu.au

Abstract. Recent algebraic attacks on LFSR-based stream ciphers and S-boxes have generated much interest as they appear to be extremely powerful. Theoretical work has been developed focusing around the Boolean function case. In this paper, we generalize this theory to arbitrary finite fields and extend the theory of annihilators and ideals introduced at Eurocrypt 2004 by Meier, Pasalic and Carlet. In particular, we prove that for any function f in the multivariate polynomial ring over $GF(q)$, f has a low degree multiple precisely when two low degree functions appear in the same coset of the annihilator of $f^{q-1}-1$. In this case, many such low degree multiples exist.

Keywords: Algebraic attacks, stream ciphers, finite fields, annihilator.

1 Introduction

Algebraic attacks on stream ciphers based on linear feedback shift registers and S-boxes have been studied in [1, 2, 3, 4, 5, 8]. Such shift registers (LFSRs) comprise a linear part L and a non-linear combining function f. The problem of finding plaintext bits given ciphertext output is thus reduced to finding solutions to systems of equations involving L and f. The function f is, in practice, of high degree, making standard techniques for solving a given system of equations too time-consuming to be of practical interest. The recent algebraic attacks find a multiple of f which is of low degree; however, this multiple is determined on an ad-hoc basis, [4,5]. The paper by Meier, Pasalic and Carlet [8] stream-lines the ideas developed behind these attacks, reducing and simplifying the various scenarios which have so far been considered. The authors also propose an algorithm to decide whether a Boolean function has low degree multiples.

In this paper, we completely generalize the work of [8] to the case of $GF(q)$ for any prime power q. We further develop the work on annihilators and ideals and show how the ideal structure assists in determining low degree multiples. Since a number of cryptographic systems employ fields other than $GF(2)$ (for instance, SOBER-t16, SOBER-t32, Camellia, AES) it is important to be able to analyze their complete structure regardless of the field used.

* Supported by a Discovery Grant of the Australian Research Council. The author wishes to thank COSIC/ESAT at KULeuven for their hospitality February - June 2004, where she was a Visiting Professor.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 84–91, 2004.

We show that, as for the Boolean case, the S3 scenarios introduced in [4] can be essentially reduced to one case over $GF(q)$ using the interplay between the multivariate function f and its complementary function $f^{q-1}-1$ over $GF(q)$. In Theorem 1 below, we generalize the result of [4,5] to show that any function f over $GF(q)$ has a low degree multiplier, that is, a low degree function g such $fg = 0$ or fg has low degree.

In Section 3, we examine the annihilator sets of the functions f and $f^{q-1}-1$ in order to gain deeper insight into their structure in the polynomial ring. This leads to an analysis of the coset structure of the annihilator ideals in Section 4 yielding the result (Theorem 4) that low degree multiples of f ($fg = h$ is of low degree) exist precisely when some coset of $An(f^{q-1}-1)$ contains more than one low degree function. By appropriately identifying coset representatives, a simple count for any fixed degree d will verify whether or not this coset property holds. This result is also new in the Boolean case.

2 The S3 Scenarios

In [4] the authors consider the situation of reducing the degree of a Boolean function by multiplying it by another non-zero function and present three cases, each of which has a method of solution. In each of these cases, f is considered to be a function of high degree. The cases are as follows:

S3a. The function g has low degree and $fg \neq 0$ has low degree.
S3b. The function g has low degree and $fg = 0$.
S3c. The function g has high degree and $fg \neq 0$ has low degree.

The words 'high' and 'low' are not defined explicitly, but in [5] theory indicates that low degree for a Boolean function in n variables means that the degree is less than about $n/2$.

In [8], the authors point out that these three cases can essentially be reduced to one situation as follows. In S3c, let $fg = h$. Then $h = fg = f^2g$ (since $f = f^2$ over $GF(2)$) $= f(fg) = fh$, and so we are in case S3a.

The following lemma re-organizes Proposition 1 of [8] and the additional comments in section 3 of that paper.

Lemma 1. *Suppose f is a Boolean function of high degree. If f is in category S3a, then $f+1$ is in category S3b. If f is in category S3b, then $f+1$ is in category S3a.*

Proof. Let f be in category S3a. So there exist low degree functions g and h such that $fg = h \neq 0$. Then $fg = f^2g = f(fg)$ implies $(f+1)fg = (f+1)h = 0$. So $f+1$ is in category S3b.

Now suppose f is in category S3b. Then there exists a non-zero, low degree function g such that $fg = 0$. Then $(f+1)g = g$ and so $f+1$ is in category S3a.□

One of the purposes of the current article is to generalize many of the existing results on algebraic attacks to general finite fields. The categories S3a, S3b and

S3c apply equally to any function $f: GF(q)^n \to GF(q)$, q a prime power. We show in the next two lemmas that a reduction to one case also applies here.

We note first of all that

$$(f^{q-1}-1)^q = f^{q-1}-1 = f^{q-1}-f+f-1 = (f+1)(f^{q-1}-1)$$

and so $(f^{q-1}-1)^{q-1}-1 = f$. This fact, along with $f(f^{q-1}-1)=0$, leads us to call $f^{q-1}-1$ the *orthogonal complement*, or simply, the *complementary function* of f.

Lemma 2. *Let $f: GF(q)^n \to GF(q)$ be in category S3c. Then $f^{q-1}-1$ is in category S3b. If $f^{q-1}-1$ is in category S3b, then f is in S3c.*

Proof. Let $fg = h$, g of high degree and $h \neq 0$ of low degree. Then $h = fg = f^q g = f^{q-1}(fg) = f^{q-1}h$. Thus $(f^{q-1}-1)h = 0$. The second part follows from the remark preceding Lemma 2. □

Lemma 2 indicates that the perceived need to look for low degree annihilators is an illusion. It suffices to find any annihilators, and alternate between f and $f^{q-1}-1$.

Lemma 3. *Suppose $f: GF(q)^n \to GF(q)$. Then f is in category S3a if and only if $f^{q-1}-1$ is in category S3b. So f is in category S3b if and only if $f^{q-1}-1$ is in category S3a.*

Proof. Suppose f is in category S3a. So there exist low degree functions g and h such that $fg = h \neq 0$. Then $fg = f^q g = f^{q-1}(fg)$ implies $(f^{q-1}-1)h = 0$ and so $f^{q-1}-1$ is in category S3b. Suppose f is in category S3b. So there exists a non-zero, low degree function g such that $fg = 0$. Then $(f^{q-1}-1)g = f^{q-2}(fg) - g = -g$ and so $f^{q-1}-1$ is in category S3a. The remainder follows since $(f^{q-1}-1)^{q-1}-1 = f$. □

In [4] the following theorem is proved, thus providing the existence of a suitable multiplier g for the cases S3a and S3b:

Theorem 6.0.1 [4,5] Let $f: GF(2)^n \to GF(2)$ be any Boolean function. Then there is a Boolean function $g \neq 0$ of degree at most $\lceil n/2 \rceil$ such that $fg = 0$ or has degree at most $\lfloor n/2 \rfloor$.

We now generalize this result to $GF(q)$.

Theorem 1. *Let $f: GF(q)^n \to GF(q)$. Then there is a function $g: GF(q)^n \to GF(q)$, $g \neq 0$, $deg(g) \leq \lceil (q-1)n/2 \rceil$ such that either $fg = 0$ or $deg(fg) \leq \lfloor (q-1)n/2 \rfloor$.*

Proof. Note that the maximum degree of any monomial over $GF(q)^n$ is $(q-1)n$. Also, the number of such monomials is q^n which is therefore the dimension of the vector space they generate. Let A be the set of all monomials of degree $\leq \lfloor (q-1)n/2 \rfloor$. Let B be the set of all products of f by monomials of degee $\leq \lceil (q-1)n/2 \rceil$. Then

$$|A| + |B| = \sum_{i=0}^{\lfloor (q-1)n/2 \rfloor} \binom{(q-1)n}{i} + \sum_{i=0}^{\lceil (q-1)n/2 \rceil} \binom{(q-1)n}{i}$$

$$= \sum_{i=0}^{\lfloor (q-1)n/2 \rfloor} \binom{(q-1)n}{i} + \sum_{\lfloor (q-1)n/2 \rfloor}^{(q-1)n} \binom{(q-1)n}{i}$$

$$= \sum_{i=0}^{(q-1)n} \binom{(q-1)n}{i} + \binom{(q-1)n}{\lfloor (q-1)n/2 \rfloor} > 2^{(q-1)n} \geq q^n.$$

Since $|A| + |B|$ is larger than the dimension of the vector space generated by the monomials, we have linear dependencies in $A \cup B$. Since all elements of A are independent, it follows that there exist elements $a_j \in A$ and fb, $fb_i \in B$ such that for some finite sums, $fb = \sum_i fb_i + \sum_j a_j$. This produces a non-zero function g of the desired degree such that either $fg = 0$ or $deg(fg) \leq \lfloor (q-1)n/2 \rfloor$. □

Meier et al. [8] introduce the notion of algebraic immunity for Boolean functions. Here we generalize the notion.

Definition. The ***algebraic immunity*** of the function $f : GF(q)^n \to GF(q)$ is $AI(f) = min\ \{d \geq 1 |\ deg(g) = d,\ g : GF(q)^n \to GF(q)$ and either $gf = 0$ or $g(f^{q-1} - 1) = 0\}$.

Theorem 1 along with Lemma 3 shows that the algebraic immunity of any such function f is at most $\lfloor (q-1)n/2 \rfloor$.

3 The Annihilator Set

For f in the ring $R = F[x_1, x_2, \ldots x_n]$, $F = GF(q)$, the product $f(f^{q-1} - 1) = 0$. Thus

$$An(f) = \{g \in R | fg = 0\}$$

contains the function $f^{q-1} - 1$. Following Meier et al. [8], we refer to $An(f)$ as the ***annihilator set*** of f in the ring R.

It is easy to verify that $An(f)$ is an ideal in the ring R generated by the element $f^{q-1} - 1$ and is therefore a principal ideal [7] which we denote by $\langle f^{q-1} - 1 \rangle$. If q is odd, this ideal is never maximal as $f^{q-1} - 1 = \left(f^{\frac{q-1}{2}} - 1\right)\left(f^{\frac{q-1}{2}} + 1\right)$ and so $< f^{q-1} - 1 > \subsetneq \langle f^{\frac{q-1}{2}} - 1 \rangle,\ \langle f^{\frac{q-1}{2}} + 1 \rangle$.

We consider the question of the size $|An(f)|$ of $An(f)$. Consider the equation $fg = 0$. Since for any vector x, g may take only the zero value at any non-zero position of f, this leaves g to take on any values possible where f is zero. Let $S_i(f)$ be the number of positions at which f is i, computed over all vectors $x = (x_1, x_2, \ldots x_n) \in GF(q)^n$. Then $q^{S_0(f)}$ is the number of ways that g can be chosen. If, in addition, f is balanced, that is, each value of $\{0, 1, \ldots, q^{n-1}\}$ occurs an equal number of times, then $S_0(f) = q^{n-1}$. We summarize the above remarks in the following.

Theorem 2. *The set $An(f)$ for $f \in R$ is a principal ideal of R generated by $f^{q-1}-1$. Moreover, letting $S_0(f)$ be the number of positions at which f is zero, then $|An(f)| = q^{S_0(f)}$. In addition, if f is balanced, then $|An(f)| = q^{q^{n-1}}$.*

Over $GF(2)$, if f is balanced, so is $f^{q-1}-1 = f+1$. However, for $GF(q)$, $q > 2$, $f^{q-1}-1$ is never balanced, since it only takes on the values 0 (for $f = 0$) and -1 (for all other values of f).

Before considering the quotient ring $R/An(f)$ in more depth, we consider relationships between various annihilator ideals in the following lemma.

Lemma 4. *For non-zero functions $f^{kl}-i$ and f^k-j we have $An(f^{kl}-i) \cap An(f^k-j) = \{0\}$ if $i \neq j^l$ for all $i, j \in GF(q)$, k and l integers.*

Proof. Let $g \in An(f^{kl}-i) \cap An(f^k-j)$. Then $gi = gf^{kl}$ and $gj = gf^k$. So $gi = gj^l$ yielding $g(i-j^l) = 0$. Since $i \neq j^l$ by assumption, $i-j^l$ is an invertible element, leaving $g = 0$. □

Example 1. In $GF(5)$, $f^4+4 = (f+2)(f+3)(f+4)$ while $f^3+2 = (f+3)(f^2+2f+4)$ and so $An(f+3) \subseteq An(f^3+2)$, $An(f^4+4)$ demonstrating that generalising Lemma 4 to all powers of f is not possible.

Theorem 3. *Let c_f be the number of additive cosets of the group structure of $R/An(f) = \{r + An(f)\}$. Then $c_f = q^{q^n - S_0(f)}$ and if f is balanced, $c_f = q^{q^{n-1}(q-1)}$.*

Moreover, for any functions g_1 and $g_2 \in R$, $fg_1 = fg_2$ if and only if $g_1 + An(f) = g_2 + An(f)$.

Proof. The value of c_f comes from $c_f = |R/An(f)| = q^{q^n}/q^{S_0(f)}$ by Theorem 2 and elementary group theory [7].

Consider g_1 and g_2 in R. Then $f(g_1-g_2) = 0$ if and only if $g_1-g_2 \in An(f)$. It follows that $g_1+An(f) = g_2+An(f)$ if and only if $fg_1 = fg_2$. □

Corollary 1. *For any $f \in R$ the number of different values of fg for $g \in R$ is precisely c_f. Moreover, for any $g \notin An(f)$, the cosets $ig+An(f)$ and $jg+An(f)$, for $i \neq j$ in $GF(q)$, are distinct.*

Proof. The first statement follows directly from the theorem.
Consider $ig + An(f) = jg + An(f)$, $g \notin An(f)$, $i \neq j$, $i, j \in GF(q)$. Then by Theorem 3, $ig-jg = (i-j)g \in An(f)$, so $(i-j)gf = 0$. But $i-j$ is invertible and so $gf = 0$ putting $g \in An(f)$ which is a contradiction. □

Corollary 2. *For functions f, g and h, $g \notin An(f)$, if $h + An(f) \neq ig + An(f)$ for any $i \in GF(q)$, then $jh + An(f) \neq kg + An(f)$ for any j, $k \in GF(q)^*$.*

Proof. Suppose $jh + An(f) = kg + An(f)$ for some i and j in $GF(q)^*$. Then $j^{-1}(jg + An(f)) = h + An(f) = (j^{-1}k)g + An(f)$ which is a contradiction. □

Corollary 3. *For any function f of R, $q-1|c_f - 1$.*

Proof. This follows from Corollaries 1 and 2.

We note that the algorithms of [8] can be easily adapted to the general case $GF(q)$. □

4 The Cosets

The upper bound given in Theorem 1 has been shown to be a best bound in [6]. We therefore concentrate in this section on possible degrees of the resultant (non-zero) product fg. In view of Lemma 2, in fact, it suffices to find any function g such that fg is either zero or of low degree. We first determine appropriate coset representatives for the cosets of $R/An(f^{q-1}-1) = R/\langle f\rangle$, where $f^{q-1}-1$ is fixed and non-zero. We use $c_{f^{q-1}-1}$ to denote the number of such cosets.

Choose a function r of least degree in any non-zero coset. Then every element in the coset is of the form $r+fg$ for some $g \in R$. From henceforth, unless otherwise stated, we assume that the coset representative r is of least degree in the coset.

Theorem 4. *The function f is in category S3a precisely when some coset of $An(f^{q-1}-1)$ contains two distinct elements of low degree. In this case, R contains at least $|An(f)|$ multiples of f of low degree.*

Proof. Suppose f is in category S3a. Let $fg = h \neq 0$ where g and h are of low degree, say $d = max\{deg(g), deg(h)\}$. Then $g + An(f^{q-1}-1) = g + h + An(f^{q-1}-1)$ contains both g and $g+h \neq g$.

Now let h_1 and h_2 be (low) degree d elements of $r + An(f^{q-1}-1)$. Then $0 \neq h = h_1 - h_2 \in \langle f\rangle$, and so $h = fg$ for some $g \in R$ and the degree of h is $\leq d$. Consider the element g, which belongs to a coset of $An(f)$. By Theorem 3, for any g‘ in this coset, $fg = fg$‘. The size of this coset is $|An(f)|$.

It remains to show that f is in category S3a. But $h = fg = f^q g = f^{q-1}h$ implies $h(f^{q-1}-1) = 0$ and so $f^{q-1}-1$ is in S3b. By Lemma 3, $(f^{q-1}-1)^{q-1}-1 = f$ is in category S3a. □

Lemma 3 and the identity $(f^{q-1}-1)^{q-1} - 1 = f$ produce the analogue theorem:

Theorem 4′. *The function $f^{q-1}-1$ is in category S3a precisely when some coset of $An(f)$ contains two distinct elements of low degree. In this case, R contains at least $|An(f^{q-1}-1)|$ multiples of $f^{q-1}-1$ of low degree.*

Note that Theorems 4 and 4′ actually show more than is stated: if a coset of $An(f^{q-1}-1)$ contains two functions of degree $\leq d$, then f has a multiple of degree $\leq d$. Analogously for $An(f)$.

Implications of Theorems 4 and 4′ on the Choice of f.
Consider the distribution of low degree functions across cosets. There are relatively few constant or linear functions and the chances of them being widely distributed in cosets is high. In fact, constants are always distributed well according to Corollary 1 of Theorem 3. Over Z_2, given a key of size n, there are

$2^{d+1}n^{d(d+1)/2}$ polynomials of degree $\leq n$ out of a total of $2^{n+1}n^{n(n+1)/2}$ possible polynomials. One would therefore expect, for d rather small and n large, to see the possibility of the existence of annihilators go to zero as n goes to infinity as demonstrated in [8].

However, once we find two low degree elements of a coset, Theorem 4 indicates that many low degree multiples of f are available, and so the chance of finding such a multiple is rather good.

Thus, when choosing f, it is essential to verify that no cosets of $An(f)$ or of $An(f^{q-1}-1)$ contain several low degree functions. If the cosets are assigned least degree coset representatives, we can formulate a design criterion as follows:

Criterion. No coset of $An(f)$ or of $An(f^{q-1}-1)$ with low degree representative should contain a second low degree function.

In practice, degree 6 still appears to be sufficiently high and so it suffices to check cosets with representatives of degree ≤ 5 [8]. In this case, simply verifying that the number of coset representatives of degree ≤ 5 is precisely the total number of functions of degree ≤ 5 is sufficient to verify the above Criterion.

Cosets of $An(f)$ do not act independently in attracting low degree functions. The following lemma demonstrates this.

Lemma 5. *The coset $An(f)$ of R contains two degree $\leq d$ functions if and only if every coset $r + An(f)$, deg $r \leq d$, contains two functions of degree $\leq d$.*

Proof. Suppose g_1, $g_2 \in An(f)$ have degree $\leq d$. Then $r + An(f) = r + g_1 + An(f) = r+g_2+An(f)$ and $deg\ (r+g_1)$, $deg\ (r+g_2) \leq d$ while $r+g_1 \neq r+g_2$. The converse is trivial. □

It follows that the Criterion above is equivalent to

Criterion*. Neither $An(f)$ nor $An(f^{q-1}-1)$ should contain two low degree functions.

Example 2. We finish with a small example showing how the annihilator cosets can vary the distribution of low degree functions depending on the choice of f. We let $R = GF(2)[x, y]$ be the multivariate ring of two variables over $GF(2)$. So $R = \{a + bx + cy + dxy \mid a, b, c, d \in GF(2)\}$ has 2^4 elements.

Let $f = x+y+xy$. Then $An(f+1) = \{0, x, y, x+y, xy, x+xy, y+xy, x+y+xy\}$ has just two cosets in R. The second is $1 + An(f+1)$. These two cosets separate degree zero functions (in fact, Corollary 1 to Theorem 3 shows that degree zero functions are always separated if the ideal is not the whole ring), but do not separate degree 1 functions. Hence, f will have a degree 1 multiple. In fact, Theorem 4 shows that some multiple of f is $x + (x + y) = y$.

Let $g = x + xy$. Then $An(g + 1) = \langle g \rangle = \{0, x + xy\}$. Thus $An(g + 1)$ has eight cosets in R. These are, in addition to the above, $\{1, 1 + x + xy\}$, $\{x, xy\}$, $\{y, x + y + xy\}$, $\{x + y, y + xy\}$, $\{1 + x + y, 1 + y + xy\}$, $\{1 + x, 1 + xy\}$, $\{1 + y, 1 + x + y + xy\}$. This ideal therefore separates out all of the degree 1 and all of the degree 2 functions.

References

1. F. Armknecht and M. Krause. Algebraic attacks on stream combiners with memory. In *Advances in Cryptology - CRYPTO 2003*, volume LNCS 2729, pages 162 - 176. Springer-Verlag, 2003.
2. N. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003*, volume LNCS 2729, pages 176 -194. Springer-Verlag, 2003.
3. N. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - ASIACRYPT 2002*, volume LNCS 2501, pages 267 - 287. Springer-Verlag, 2002.
4. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - EUROCRYPT 2003*, volume LNCS 2656, pages 346 - 359. Springer-Verlag, 2003.
5. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. Extended version of [8], available at http://www.cryptosystem.net.stream/, 2003.
6. J.-CH. Faugère and G. Ars. An algebraic cryptoanalysis of nonlinear filter generators using Gröbner bases. Available on the web, 2003. http://www.inria.fr/rrrt/rr-4739.html.
7. S. Maclane and G. Birkhoff. *Algebra*. MacMillan, New York, 1967.
8. W. Meier, E. Pasalic and C. Carlet. Algebraic attacks and decomposition of boolean functions. In *Advances in Cryptology - EUROCRYPT 2004*, volume LNCS 3027, pages 474 - 491. Springer-Verlag, 2004.

Results on Algebraic Immunity for Cryptographically Significant Boolean Functions

Deepak Kumar Dalai, Kishan Chand Gupta, and Subhamoy Maitra

Applied Statistics Unit, Indian Statistical Institute,
203, B T Road, Calcutta 700 108, India
{deepak_r, kishan_t, subho}@isical.ac.in

Abstract. Recently algebraic attack has received a lot of attention in cryptographic literature. It has been observed that a Boolean function f, interpreted as a multivariate polynomial over $GF(2)$, should not have low degree multiples when used as a cryptographic primitive. In this paper we show that high nonlinearity is a necessary condition to resist algebraic attack and explain how the Walsh spectra values are related to the algebraic immunity (resistance against algebraic attack) of a Boolean function. Next we present enumeration results on linearly independent annihilators. We also study certain classes of highly nonlinear resilient Boolean functions for their algebraic immunity.

Keywords: Algebraic Attacks, Annihilators, Boolean Functions, Nonlinearity, Walsh Spectra.

1 Introduction

A very well studied model of stream cipher is the nonlinear combiner model, where the outputs of several LFSRs are combined using a nonlinear Boolean function to produce the key stream. This model has undergone a lot of cryptanalysis and to resist those attacks, different design criteria have been proposed for both the LFSRs and the combining Boolean function. There are large number of important papers in this direction and one may refer to [1, 11, 2, 23] and the references in these papers for more details. Very recently a new attack has gained lot of attention that uses over defined systems of multivariate linear equations to recover the secret key and it is known as algebraic attack [4, 5, 6, 13].

Given a Boolean function f on n-variables, different kinds of scenarios related to low degree multiples of f have been studied in [5, 13]. The core of the analysis is to find out minimum (or low) degree annihilators of f and $1 + f$, i.e., to find out minimum (or low) degree functions g_1, g_2 such that $f * g_1 = 0$ and $(1 + f) * g_2 = 0$. To mount the algebraic attack, one needs only the low degree linearly independent annihilators [5, 13] of $f, 1 + f$.

In this paper we refer the immunity of a Boolean function against algebraic attack as algebraic immunity. We study the relationship between algebraic immunity and nonlinearity of a Boolean function. We show that a Boolean function

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 92–106, 2004.

with low nonlinearity will have low algebraic immunity. The result relates the algebraic immunity to the Walsh spectra of a Boolean function. We also present enumeration results on number of such annihilators.

It is known that a Boolean function must be resilient, should have high nonlinearity and algebraic degree to be used in the nonlinear combiner model of stream cipher. We study such functions for their algebraic immunity. We present experimental results on highly nonlinear resilient functions which are rotation symmetric [8, 20, 21, 10, 12]. The experiments have been done using Algorithm 1 [13] on functions of 7, 8 and 9 variables and their complements. The results found are encouraging, which shows that there are highly nonlinear resilient functions which are also optimal in terms of their algebraic immunity. Further we study different construction methods of resilient functions. We note that the Siegenthaler's construction [19] is not good in terms of algebraic immunity. On the other hand we show that the construction presented in [14] (basically a construction similar to the Tarannikov's construction [22]) is encouraging in terms of algebraic immunity. We have also experimentally studied some functions which are of Maiorana-McFarland type [16], i.e., which can be seen as concatenation of affine functions.

2 Preliminaries

A Boolean function on n variables may be viewed as a mapping from $V_n = \{0,1\}^n$ into $V_1 = \{0,1\}$ and define B_n as the set of all n-variable Boolean functions. One of the standard representation of a Boolean function $f(x_1, \ldots, x_n)$ is by the output column of its *truth table*, i.e., a binary string of length 2^n,

$$f = [f(0,0,\ldots,0), f(1,0,\ldots,0), f(0,1,\ldots,0), \ldots, f(1,1,\ldots,1)].$$

The set of $x \in V_n$ for which $f(x) = 1$ (respectively $f(x) = 0$) is called the on set (respectively off set), denoted by 1_f (respectively 0_f). We say that a Boolean function f is balanced if the truth table contains an equal number of 1's and 0's.

The Hamming weight of a binary string S is the number of ones in the string. This number is denoted by $wt(S)$. The Hamming distance between two strings, S_1 and S_2 is denoted by $d(S_1, S_2)$ and is the number of places where S_1 and S_2 differ. Note that $d(S_1, S_2) = wt(S_1 + S_2)$ (by abuse of notation, we also use $+$ to denote the $GF(2)$ addition, i.e., the XOR).

Any Boolean function has a unique representation as a multivariate polynomial over $GF(2)$, called the algebraic normal form (ANF),

$$f(x_1, \ldots, x_n) = a_0 + \sum_{1 \le i \le n} a_i x_i + \sum_{1 \le i < j \le n} a_{ij} x_i x_j + \ldots + a_{12\ldots n} x_1 x_2 \ldots x_n,$$

where the coefficients $a_0, a_{ij}, \ldots, a_{12\ldots n} \in \{0,1\}$. The algebraic degree, $\deg(f)$, is the number of variables in the highest order term with non zero coefficient. A Boolean function is affine if there exists no term of degree > 1 in the ANF and the set of all affine functions is denoted $A(n)$. An affine function with constant term equal to zero is called a linear function.

It is known that a Boolean function should be of high algebraic degree to be cryptographically secure [7]. Further, it has been identified recently, that it should not have a low degree multiple [5]. The algebraic attack (see [5, 13] and the references in these papers) is getting a lot of attention recently. To resist algebraic attacks, the Boolean functions used in the cryptosystems should be chosen properly. It is shown [5] that given any n-variable Boolean function f, it is always possible to get a Boolean function g with degree at most $\lceil \frac{n}{2} \rceil$ such that $f * g$ is of degree at most $\lceil \frac{n}{2} \rceil$. Here the functions are considered to be multivariate polynomials over $GF(2)$ and $f * g$ is the polynomial multiplication over $GF(2)$. Thus while choosing an f, the cryptosystem designer should be careful that it should not happen that degree of $f * g$ falls much below $\lceil \frac{n}{2} \rceil$. At this point we present two important issues related to algebraic attack [5, 13].

1. Take $f, g, h \in B_n$. Assume that there exists a nonzero function g of low degree such that $f * g = h$, where h is a nonzero function of low degree and without loss of generality, $\deg(g) \leq \deg(h)$. This is because, if $\deg(g) > \deg(h)$, then $f * h = f * f * g = f * g = h$, so one can use h in place of g.
2. Assume there exists a nonzero function g of low degree such that $f * g = 0$. This g is called the annihilator of f.

The following two results from [5, 13] are relevant here.

1. Let $f, g \in B_n$. Then g is an annihilator of f iff $1_f \subseteq 0_g$.
2. Let $f \in B_n$. Then there is a non zero $g \in B_n$ of degree $\leq \lceil \frac{n}{2} \rceil$ such that $f * g$ is of degree $\leq \lceil \frac{n}{2} \rceil$.

We will update the notion a little bit for our purpose where we will consider the multiples of both f and $1 + f$.

1. Take $f, g, h \in B_n$. Assume that there exists a nonzero function g of low degree such that $f * g = h$ or $(1 + f) * g = h$, where h is a nonzero function of low degree and without loss of generality, $\deg(g) \leq \deg(h)$. Among all such h's we denote the lowest degree h (may be more than one and then we take any one of them) by $ldgm_n(f)$.
2. Assume there exists a nonzero function g of low degree such that $f * g = 0$ or $(1 + f) * g = 0$. Among all such g's we denote the lowest degree g (may be more than one and then we take any one of them) by $ldga_n(f)$.

From the discussion in [13], it can be deduced that for $f \in B_n$, $\deg(ldgm_n(f)) = \deg(ldga_n(f))$. Keeping this in mind, we present the following definition of algebraic immunity.

Definition 1. *The algebraic immunity of an n-variable Boolean function f is denoted by $\mathcal{AI}_n(f)$ which is basically* $\deg(ldgm_n(f))$ *or* $\deg(ldga_n(f))$.

The nonlinearity of an n-variable function f is the minimum distance from the set of all n-variable affine functions, i.e.,

$$nl(f) = \min_{g \in A(n)} (d(f, g)).$$

Boolean functions used in crypto systems must have high nonlinearity to prevent linear attacks [7].

It is known that there are highly nonlinear Boolean functions of low degree, as example there exist quadratic bent functions, which are of degree 2 and maximum possible nonlinearity $2^{n-1} - 2^{\frac{n}{2}-1}$, when n is even. Such functions f, as they are by themselves of low algebraic degree, will have low values of algebraic immunity $\mathcal{AI}_n(f)$ (see Definition 1 later). On the other hand, we may have Boolean functions of low nonlinearity with high algebraic degree. Interestingly this is not the case in terms of algebraic immunity. In this paper we show that if a function is of low nonlinearity, then it must have a low value of $\mathcal{AI}_n(f)$. This implies that if one chooses a function with good value of $\mathcal{AI}_n(f)$, that will automatically provide a good nonlinearity. That is the algebraic immunity property takes care of two fundamental properties of a Boolean function, algebraic degree and nonlinearity, at the same time. Further we will show that this property stays almost unchanged with respect to linear transformation unlike correlation immunity or propagation characteristics.

Many properties of Boolean functions can be described by the Walsh transform. Let $x = (x_1, \ldots, x_n)$ and $\omega = (\omega_1, \ldots, \omega_n)$ both belonging to $\{0,1\}^n$ and $x \cdot \omega = x_1\omega_1 + \ldots + x_n\omega_n$. Let $f(x)$ be a Boolean function on n variables. Then the *Walsh transform* of $f(x)$ is an integer valued function over $\{0,1\}^n$ which is defined as

$$W_f(\omega) = \sum_{x \in \{0,1\}^n} (-1)^{f(x)+x\cdot\omega}.$$

A Boolean function f is balanced iff $W_f(0) = 0$. The nonlinearity of f is given by $nl(f) = 2^{n-1} - \frac{1}{2}\max_{\omega\in\{0,1\}^n} |W_f(\omega)|$. Correlation immune functions and resilient functions are two important classes of Boolean functions. A function is m-resilient (respectively mth order correlation immune) iff its Walsh transform satisfies

$$W_f(\omega) = 0, \text{ for } 0 \le wt(\omega) \le m \text{ (respectively } 1 \le wt(\omega) \le m).$$

Following the notation as in [16, 17, 21] we use (n, m, d, σ) to denote n-variable, m-resilient function with degree d and nonlinearity σ. Further, by $[n, m, d, \sigma]$ we denote unbalanced n-variable, mth order correlation immune function with degree d and nonlinearity σ.

3 On Algebraic Attacks

Towards proving the results relating algebraic immunity and the nonlinearity of a Boolean function, we first present the following result where we relate the algebraic degree with the weight of the function.

Theorem 1. *Let $f \in B_n$ and $AI_n(f) > d$. Then*

$$\sum_{i=0}^{d} \binom{n}{i} \le wt(f) \le \sum_{i=0}^{n-(d+1)} \binom{n}{i}.$$

Proof. Consider that f has an annihilator g of degree d. Let the ANF of $g = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{1 \le i < j \le n} a_{i,j} x_i x_j + ... + \sum_{1 \le i_1 \le ... \le i_d \le n} a_{i_1,...i_d} x_{i_1} ... x_{i_d}$. Note that $f(x) = 1$ implies $g(x) = 0$. So, we will be able to get linear equations from $g(x) = 0$ on the a's in ANF of g. That is we will get $wt(f)$ many homogeneous linear equations on the a's.

Solving the system of linear homogeneous equations, we can find out annihilators g of degree $\le d$ on nontrivial solutions. (In case of a trivial solution we will get all the a's equal to zero, i.e., $g(x) = 0$, which is not acceptable as we are interested in non zero $g(x)$.)

Here, we have $\sum_{i=0}^{d} \binom{n}{i}$ number of variables (the a's for the monomials up to degree d) and $wt(f)$ many number of equations. If the number of variables is greater than number of equations then we will get nontrivial solutions. Thus f has no annihilator g of degree d implies the number of equations is greater than or equal the number of variables. So, there must be at least $\sum_{i=0}^{d} \binom{n}{i}$ number of equations, i.e., $wt(f) \ge \sum_{i=0}^{d} \binom{n}{i}$. Similarly, when considering $1 + f$, we get $wt(1 + f) \ge \sum_{i=0}^{d} \binom{n}{i}$. This gives, $wt(f) \le 2^n - \sum_{i=0}^{d} \binom{n}{i}$, i.e., $wt(f) \le \sum_{i=0}^{n-(d+1)} \binom{n}{i}$. □

Theorem 1 also gives an alternative proof of $AI_n(f) \le \lceil \frac{n}{2} \rceil$ which was given in [5]. For any f the inequality in Theorem 1 will not be satisfied if $d > n - (d + 1) \Rightarrow d > \frac{n-1}{2} \Rightarrow d \ge \lceil \frac{n}{2} \rceil$. That is, for any f the inequality in Theorem 1 will not be satisfied if $AI_n(f) > d \ge \lceil \frac{n}{2} \rceil$.

However, the reverse direction of Theorem 1 is not always true. For example, the affine functions are balanced, but clearly they have linear annihilators.

Based on Theorem 1, the following result gives a bound on $wt(f)$, where f does not have multiples of degree less than $\lceil \frac{n}{2} \rceil$.

Corollary 1. *$AI_n(f) = \lceil \frac{n}{2} \rceil$ implies*

1. *f is balanced when n is odd*
2. *$\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i} \le wt(f) \le \sum_{i=0}^{\frac{n}{2}} \binom{n}{i}$ when n is even.*

Proof. The $wt(f)$ will satisfy Theorem 1 for $d = \lceil \frac{n}{2} \rceil - 1$. That is

$$\sum_{i=0}^{\lceil \frac{n}{2} \rceil - 1} \binom{n}{i} \le wt(f) \le \sum_{i=0}^{n - \lceil \frac{n}{2} \rceil} \binom{n}{i} \Rightarrow \sum_{i=0}^{\lceil \frac{n}{2} \rceil - 1} \binom{n}{i} \le wt(f) \le \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{i}.$$

When n is odd, $\lfloor \frac{n}{2} \rfloor = \lceil \frac{n}{2} \rceil - 1$ and hence $wt(f) = \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{i} = 2^{n-1}$. For n even, $\lfloor \frac{n}{2} \rfloor = \lceil \frac{n}{2} \rceil$ and the result follows. □

Now we connect algebraic immunity with nonlinearity.

Theorem 2. *If $nl(f) < \sum_{i=0}^{d} \binom{n}{i}$, then $AI_n(f) \le d + 1$.*

Proof. Let $\alpha \in \{0,1\}^n$ such that $|W_f(\alpha)|$ is maximum, i.e., $nl(f) = \min\{wt(f + \alpha \cdot x), wt(1 + f + \alpha \cdot x)\}$. We use the contrapositive result of Theorem 1. If d is

the minimum integer such that $\min\{wt(f+\alpha\cdot x), wt(1+f+\alpha\cdot x)\} < \sum_{i=0}^{d}\binom{n}{i}$ then $AI_n(f+\alpha\cdot x) \leq d$.

Let $F \in B_n$ and $l \in A_n$. For any g such that $F*g = 0$, $(F+l)*((l+1)*g) = 0$. Similarly for any g such that $(1+F)*g = 0$, $(1+F+l)*((l+1)*g) = 0$. Hence $AI_n(F+l) \leq AI_n(F)+1$. Since $f = (f+\alpha\cdot x)+\alpha\cdot x$, $AI_n((f+\alpha\cdot x)+\alpha\cdot x) \leq AI_n(f+\alpha\cdot x)+1 \leq d+1$. □

From the above theorem we directly get the following result.

Corollary 2. *If $AI_n(f) > d+1$ then $nl(f) \geq \sum_{i=0}^{d}\binom{n}{i}$.*

In Theorem 2, the interesting situation is when $\deg(f) > d+1$. Because, when $\deg(f) \leq d+1$, then irrespective of the nonlinearity of f, $AI_n(f) \leq d+1$, since $f*(1+f) = 0$. Since there are low degree functions with very high nonlinearity (as example quadratic bent function), it is clear that there are functions f with high nonlinearity and low $AI_n(f)$ (basically $1+f$). On the other hand, we prove that one should not use functions f with low nonlinearity because in that case $AI_n(f)$ will be low. Thus the candidate functions f with good algebraic immunity should be of high algebraic degree as well as high nonlinearity. Then one should check the multiples at degree $AI_n(f)$ before deciding the quality of the function, i.e., in the best possible scenario $AI_n(f) = \lceil \frac{n}{2} \rceil$. A function with $AI_n(f) = \lceil \frac{n}{2} \rceil$, by itself, takes care of good nonlinearity and algebraic degree.

3.1 Count of Annihilators

In the proof of Theorem 1, we get $wt(f)$ many homogeneous linear equations using the a's. Let us denote the coefficient matrix of this system of equations by M. Then M has $wt(f)$ many rows and $\sum_{i=0}^{d}\binom{n}{i}$ many columns. The rank of the matrix M, $r \leq \min\{wt(f), \sum_{i=0}^{d}\binom{n}{i}\}$.

1. If $r = \sum_{i=0}^{d}\binom{n}{i}$, then there is no annihilator of degree $\leq d$.
2. If $r < \sum_{i=0}^{d}\binom{n}{i}$, then there are annihilators of degree $\leq d$. There will be $\sum_{i=0}^{d}\binom{n}{i} - r$ many linearly independent annihilators having degree $\leq d$.

It is clear [5] that a larger number of independent annihilators helps better in cryptanalysis. Thus when considering a Boolean function one should check the number of independent annihilators at the lowest possible degree.

Definition 2. *Given $f \in B_n$, by $\#LDA_n(f)$, we denote the number of independent annihilators of f at $AI_n(f)$.*

Theorem 3.

1. *Take $f \in B_n$, with $AI_n(f) = d+1 < \lceil \frac{n}{2} \rceil$. Then $\#LDA_n(f) \leq \binom{n}{d+1}$.*
2. *Take balanced $f \in B_n$, n even with $AI_n(f) = \frac{n}{2}$. Then $\#LDA_n(f) \geq \frac{\binom{n}{\frac{n}{2}}}{2}$.*
3. *Take a balanced function $f \in B_n$, n odd such that $AI_n(f) = \lceil \frac{n}{2} \rceil$. Then $\#LDA_n(f) = \binom{n}{\lceil \frac{n}{2} \rceil}$.*

Proof. The proof of item 1 is as follows. It is given that there is no non zero annihilator up to degree d. If one considers an annihilator of degree d, then the only solution would become the trivial zero function. The rank of the coefficient matrix M is equal to number of variables, i.e., equal to $\sum_{i=0}^{d}\binom{n}{i}$. Now the function f has annihilator at degree $d+1$. The corresponding coefficient matrix (say M') is obtained from M by adding $\binom{n}{d+1}$ columns. Thus the rank of M' will be greater or equal to the rank of M, i.e., $\sum_{i=0}^{d}\binom{n}{i}$. Number of independent solutions will be $\leq \sum_{i=0}^{d+1}\binom{n}{i} - \sum_{i=0}^{d}\binom{n}{i} = \binom{n}{d+1}$.

Now we prove item 2. Here $wt(f) = 2^{n-1}$. The function f has annihilator at degree $\frac{n}{2}$. In this case the corresponding coefficient matrix (say M) will have 2^{n-1} many rows and $\sum_{i=0}^{\frac{n}{2}}\binom{n}{i} = 2^{n-1} + \frac{\binom{n}{\frac{n}{2}}}{2}$ many columns. Thus rank of M will be $\leq 2^{n-1}$. Number of independent solutions will be $\geq (2^{n-1} + \frac{\binom{n}{\frac{n}{2}}}{2}) - 2^{n-1} = \frac{\binom{n}{\frac{n}{2}}}{2}$.

Here we present the proof of item 3. Here $wt(f) = 2^{n-1}$. It is given that there is no non zero annihilator up to degree $\lfloor \frac{n}{2} \rfloor$. If one considers an annihilator of degree $\leq \lfloor \frac{n}{2} \rfloor$, then the only solution would become the trivial zero function. In this case the number of variables (the a's) is $\sum_{i=0}^{\lfloor \frac{n}{2} \rfloor}\binom{n}{i} = 2^{n-1}$. So the coefficient matrix M is a $2^{n-1} \times 2^{n-1}$ square matrix. As it has no nontrivial solution, its rank $r = 2^{n-1}$. The function f has annihilator at degree $\lceil \frac{n}{2} \rceil$. In this case the corresponding coefficient matrix (say M') will have 2^{n-1} many rows and $2^{n-1} + \binom{n}{\lceil \frac{n}{2} \rceil}$ many columns. Thus rank of M' will be equal to that of M, i.e., 2^{n-1}. Number of independent solutions $= (2^{n-1} + \binom{n}{\lceil \frac{n}{2} \rceil}) - 2^{n-1} = \binom{n}{\lceil \frac{n}{2} \rceil}$. □

In the next section, we will study certain constructions of cryptographically significant Boolean functions in terms of algebraic immunity.

4 Studying Functions for Their Algebraic Immunity

It has been studied using statistical techniques in [13] that any randomly chosen balanced function on large number of variables will have good algebraic immunity with very high probability. This result is in a similar direction that most of the Boolean functions are of high algebraic degree or of high nonlinearity in general. That is if one chooses a Boolean function randomly, the probability that these properties will be good is high. However, when considering a specific construction technique, the number of functions constructed by that method is much lower than the total space of Boolean functions and generally such statistical analysis does not work.

4.1 Experimental Results on Rotation Symmetric Boolean Functions

Let us consider that we want to construct (n, m, d, x) functions with best possible parameters along with the best possible algebraic immunity. In this direction

we first refer to a small subset of Boolean functions, the rotation symmetric Boolean functions (RSBFs), which received a lot of attention recently [20, 21, 10, 12]. These functions are invariant under circular translation of indices in the input variables. We present experimental results related to the algebraic immunity of the RSBFs which are available in [20, 21, 10, 12].

Experiment 1: Here we test the algebraic immunity for $(7, 2, 4, 56)$ RSBFs. It is given in [20] that there are 36 such functions with $f(0) = 0$. Out of them, 24 functions contain linear terms. For these functions, $AI_n(f) = 3$, which is 1 less than highest value $\lceil \frac{n}{2} \rceil = 4$. Out of them 12 functions have $\#LDA_n(f) = 3$ and the rest 12 have $\#LDA_n(f) = 4$. The algebraic immunity of the other 12 functions, where the linear terms are not there, $AI_n(f) = 4$, which is the highest possible value. According to Theorem 3(item 3) (we have also checked by experiment), for these functions $\#LDA_n(f) = \binom{7}{\lceil \frac{7}{2} \rceil} = 35$.

Experiment 2: Here we examine the $(8, 1, 6, 116)$ RSBFs with $f(0) = 0$ which are 10272 in number [21]. Out of them, 6976 numbers attains highest algebraic immunity, i.e., 4 and we find that for these functions $\#LDA_n(f) = 35$. From Theorem 3(item 2), in this case the value should be $\geq \frac{\binom{8}{\frac{8}{2}}}{2} = 35$. Thus we find an example, where the bound is tight. For the rest $10272 - 6976 = 3296$ functions, the algebraic immunity is 3. Out of them 1536 many functions f have only one annihilator at degree 3 (but no degree 3 annihilator for $1 + f$), 1504 many functions f have no annihilator at degree 3 (but one degree 3 annihilator for $1+f$) and 256 many functions f have one annihilator at degree 3 (also one degree 3 annihilator for $1 + f$). According to Theorem 3(item 1), $\#LDA_n(f) \leq \binom{8}{3} = 56$. So for these functions, the bound is not sharp.

Experiment 3: In the above two experiments, we examined the functions which are balanced. Now we consider the $[9, 3, 5, 240]$ RSBFs which are not balanced. We consider the functions with $f(0) = 0$, and these are 8406 in number [10, 12]. According to Corollary 1(item 1), the algebraic immunity of these functions will be strictly less than 5. Here after experiment we get the algebraic immunity of all 8406 functions as 4. From Theorem 3(item 1), $\#LDA_9(f) \leq \binom{9}{4} = 126$. In the following table, we present the number of functions satisfying a particular $\#LDA_9(f)$ and $\#LDA_9(1 + f)$.

$\#LDA_9(f)$	16	17	18	19	20	21
$\#LDA_9(1 + f)$	0	1	2	3	4	5
$\#f$	5658	1758	774	180	12	24

Studying the resilient functions on 7 and 8 variables and unbalanced correlation immune functions on 9-variables for this rotation symmetric class of Boolean functions, it is evident that there exists functions which are good in terms of algebraic immunity. It will be interesting to study such functions on higher number of variables.

4.2 Analysis of Some Construction Methods

Let us start with a technical result. For notational purpose, given $f \in B_n$, we denote the set $LDGA_n(f)$ as the set of lowest degree f_1's ($f_1 \in B_n$) such that $f * f_1 = 0$ or $(1 + f) * f_1 = 0$.

Proposition 1. *Let $f, g \in B_n$ on variables $x_1, x_2, \cdots, x_n$ with $AI_n(f) = d_1$ and $AI_n(g) = d_2$. Let $h = (1 + x_{n+1})f + x_{n+1}g \in B_{n+1}$. Then*

1. *if $d_1 \neq d_2$ then $AI_{n+1}(h) = \min\{d_1, d_2\} + 1$.*
2. *Given $d_1 = d_2 = d$, $d \leq AI_{n+1}(h) \leq d + 1$. Further, $AI_{n+1}(h) = d$ iff there exists $f_1, g_1 \in B_n$ of algebraic degree d such that $\{f * f_1 = 0, g * g_1 = 0\}$ or $\{(1 + f) * f_1 = 0, (1 + g) * g_1 = 0\}$ and $\deg(f_1 + g_1) \leq d - 1$.*

Proof. Let $f_1 \in LDGA_n(f)$ and $g_1 \in LDGA_n(g)$. Thus, either $f * f_1 = 0$ which gives $(1+x_{n+1})*f_1*h = 0$ or $(1+f)*f_1 = 0$ which gives $(1+x_{n+1})*(1+f_1)*h = 0$. Also either $g * g_1 = 0$ implies $x_{n+1} * g_1 * h = 0$ or $(1 + g) * g_1 = 0$ implies $x_{n+1} * (1 + g_1) * h = 0$. Thus,

$$AI_{n+1}(h) \leq \min\{AI_n(f), AI_n(g)\} + 1. \tag{1}$$

Let $p = (1+x_{n+1})p_1 + x_{n+1}p_2 \in LDGA_{n+1}(h)$. Let us first consider the case with $h * p = 0$ which implies $(1 + x_{n+1})f * p_1 + x_{n+1}g * p_2 = 0$. So $f * p_1 = 0$ and $g * p_2 = 0$. Similarly for the case with $(1 + h) * p = 0$, i.e., $(1 + x_{n+1}) * (1 + f) * p_1 + x_{n+1}(1 + g) * p_2 = 0$, we have $(1 + f) * p_1 = 0$ and $(1 + g) * p_2 = 0$. Now there could be three cases in both the scenarios.

(a) p_1 is zero, but p_2 is non zero. So $\deg(p_2) \geq d_2$ which gives $\deg(p) \geq d_2 + 1$.
(b) p_2 is zero, but p_1 is non zero. So $\deg(p_1) \geq d_1$ which gives $\deg(p) \geq d_1 + 1$.
(c) Both p_1, p_2 are non zero. So $\deg(p_1) \geq d_1$ and $\deg(p_2) \geq d_2$, which gives $\deg(p) \geq \max\{d_1, d_2\} + 1$, when $d_1 \neq d_2$.

So for $d_1 \neq d_2$ we get,

$$AI_{n+1}(h) \geq \min\{AI_n(f), AI_n(g)\} + 1. \tag{2}$$

Equation 1, 2 give the proof of item 1.

Now we prove item 2. Consider $p = (1+x_{n+1})f_1 + x_{n+1}g_1 \in LDGA_{n+1}(h)$. It could happen that all highest degree terms of $x_{n+1}f_1 + x_{n+1}g_1$ in p get canceled and the over all degree is decreased by one. So, $d \leq AI_{n+1}(h) \leq d + 1$.

Let $AI_{n+1}(h) = d$. Then the highest degree terms of f_1 and g_1 must be same which gives $deg(f_1 + g_1) \leq d - 1$. Now we prove the other side. Let there exist $f_1, g_1 \in B_n$ of degree d such that $\deg(f_1 + g_1) \leq d - 1$ and one of the following holds

$$f * f_1 = 0, g * g_1 = 0, \tag{3}$$

$$(1 + f) * f_1 = 0, (1 + g) * g_1 = 0. \tag{4}$$

Construct $p = (1 + x_{n+1})f_1 + x_{n+1}g_1$. Thus $h * p = 0$ (when Equation 3 is considered) or $(1+h)*p = 0$ (when Equation 4 is considered). So, $AI_{n+1}(h) = d$. □

Corollary 3. *Let $f \in B_n$, $AI_n(f) = d$ and $h = x_{n+1} + f \in B_{n+1}$.*

1. *Then $d \leq AI_{n+1}(h) \leq d + 1$.*
2. *$AI_n(h) = d$ iff there exist $f_1, f_2 \in LDGA_n(f)$ such that $f * f_1 = 0$, $(1 + f) * f_2 = 0$ and $\deg(f_1 + f_2) \leq d - 1$.*

Proof. Since $x_{n+1} + f = (1 + x_{n+1})f + x_{n+1}(1 + f)$, this follows directly from Proposition 1. □

In the same line we present one more technical result.

Proposition 2. *Let $f(x_1, \ldots, x_n) \in B_n$ and $AI_n(f) = d$. Let l be a affine function with any of the following properties: (i) l is a function on $x_1, \ldots, x_n$, (ii) l is a function on variables other than $x_1, \ldots, x_n$, (iii) l is a function on $x_1, \ldots, x_n$ and some other variables. Let $l+f$ be a function on m variables. Then $d - 1 \leq AI_m(l + f) \leq d + 1$ for cases (i) and (iii) and $d \leq AI_m(l + f) \leq d + 1$ for case (ii).*

Proof. Let $g \in LDGA_n(f)$, which implies $f * g = 0$ or $(1 + f) * g = 0$ and $\deg(g) = d$. So for any affine function l, we have $(l + f) * ((1 + l) * g) = 0$ if $f*g = 0$ or $(l+f+1)*((1+l)*g) = 0$ if $(1+f)*g = 0$. Hence, $AI_m(f+l) \leq d+1$. So, the upper bound for all cases is proved.

Now we consider case (i), where l is an affine function on the variables $x_1, \ldots, x_n$. Let there be an $l \in A_n$ such that $AI_n(f + l) < d - 1$. Then $AI_n(f) = AI_n((f + l) + l) \leq AI_n(f + l) + 1 < d$, which contradicts that $AI_n(f) = d$. Thus, $AI_m(l + f) \geq d - 1$.

The lower bound of case (ii) follows from repeated application of Corollary 3.

Now we prove lower bound of case (iii). Let $l = l_1 + l_2$, where l_1 is an affine function on some or all of the variables $x_1, \ldots, x_n$ and l_2 is an affine function on some other variables. So, following case (i), we have $AI_n(f + l_1) \geq d - 1$. Then following case (ii), $AI_m(f + l) = AI_m((f + l_1) + l_2) \geq AI_m(f + l_1) = d - 1$. □

In [19] Siegenthaler proposed a construction of resilient functions. Take an initial (n, m, d, σ) function $f(x_1, \ldots, x_n)$. The function $F(x_1, \ldots, x_{n+k}) = x_{n+k} + \ldots + x_{n+1} + f(x_1, \ldots, x_n)$ will be an $(n + k, m + k, d, 2^k\sigma)$ one. From Proposition 2, we get $AI_n(f) \leq AI_{n+k}(F) \leq AI_n(f) + 1$. Thus this construction is not good in terms of algebraic immunity.

In [22], Tarannikov has proposed an important construction of resilient functions and based on that a similar kind of construction has been proposed in [14]. We will refer the construction in [14] here and study the algebraic immunity of such functions. Let us first present the construction.

An $(n, m, d, -)$ function f is called to be in *desired* form if it is of the form $f = (1 + x_n)f_1 + x_n f_2$, where f_1, f_2 are $(n - 1, m, d - 1, -)$ functions. Let f be an (n, m, d, σ) function in *desired* form, where f_1, f_2 are both $(n-1, m, d-1, -)$ functions. Let

$$F = x_{n+2} + x_{n+1} + f \text{ and}$$

$$G = (1 + x_{n+2} + x_{n+1})f_1 + (x_{n+2} + x_{n+1})f_2 + x_{n+2} + x_n.$$

In the language of [22], the function G above is said to depend quasilinearly on the pair of variables (x_{n+2}, x_{n+1}). We construct a function H in $n+3$ variables in the following way,

$$H = (1 + x_{n+3})F + x_{n+3}G.$$

Then the function H constructed from f is an $(n+3, m+2, d+1, 2^{n+1}+4\sigma)$ function in the *desired* form. Thus, this construction can be applied iteratively.

Construction 1. Let us describe this construction with some index to present the iterative effect. Let H^0 be the initial function of n variables and H^i be the constructed function after i-th iteration. Denote $H^{i'}$ as the function generated from H^i by replacing the variable x_{n+3i} by $(x_{n+3i+2} + x_{n+3i+1})$. Let $F^{i+1} = x_{n+3i+2} + x_{n+3i+1} + H^i$ and $G^{i+1} = x_{n+3i+2} + x_{n+3i} + H^{i'}$. Then the constructed function at $i+1$-th step, $H^{i+1} = (1 + x_{n+3i+3})F^{i+1} + x_{n+3i+3}G^{i+1}$.

Now we present a technical result.

Proposition 3. *For $i > 0$, $H^i = (1 + Y_i)H^0 + Y_iH^{0'} + Z_i$ where $\deg(Y_i) = i$ and $\deg(Z_i) = i + 1$.*

Proof. The base case is as follows.

$$\begin{aligned} H^1 &= (1 + x_{n+3})F^1 + x_{n+3}G^1 \\ &= (1 + x_{n+3})H^0 + x_{n+3}H^{0'} + (1 + x_{n+3})(x_{n+2} + x_{n+1}) + x_{n+3}(x_{n+2} + x_n) \\ &= (1 + Y_1)H^0 + Y_1H^{0'} + Z_1, \end{aligned}$$

where Y_1 is a 1-degree polynomial and Z_1 is a 2-degree polynomial.

Let us assume that this is true for some $k \geq 1$, i.e., $H^k = (1+Y_k)H^0 + Y_kH^{0'} + Z_k$, where Y_k is a k-degree polynomial and Z_k is $k+1$-degree polynomial. Now,

$$\begin{aligned} H^{k+1} &= (1 + x_{n+3k+3})(x_{n+3k+2} + x_{n+3k+1} + H^k) \\ &\quad + x_{n+3k+3}(x_{n+3k+2} + x_{n+3k} + H^{k'}) \\ &= (1 + x_{n+3k+3})H^k + x_{n+3k+3}H^{k'} \\ &\quad + (1 + x_{n+3k+3})(x_{n+3k+2} + x_{n+3k+1}) + x_{n+3k+3}(x_{n+3k+2} + x_{n+3k}) \\ &= (1 + x_{n+3k+3})((1 + Y_k)H^0 + Y_kH^{0'} + Z_k) \\ &\quad + x_{n+3k+3}((1 + Y_k')H^0 + Y_k'H^{0'} + Z_k') \\ &\quad + (1 + x_{n+3k+3})(x_{n+3k+2} + x_{n+3k+1}) + x_{n+3k+3}(x_{n+3k+2} + x_{n+3k}), \end{aligned}$$

where Y_k' and Z_k' are generated by replacing the variable x_{n+3k} by $(x_{n+3k+2} + x_{n+3k+1})$ in Y_k and Z_k respectively. Thus,

$$\begin{aligned} H^{k+1} &= (1 + Y_k + Y_kx_{n+3k+3} + Y_k'x_{n+3k+3})H^0 \\ &\quad + (Y_k + Y_kx_{n+3k+3} + Y_k'x_{n+3k+3})H^{0'} + (1 + x_{n+3k+3})Z_k + x_{n+3k+3}Z_k' \\ &\quad + (1 + x_{n+3k+3})(x_{n+3k+2} + x_{n+3k+1}) + x_{n+3k+3}(x_{n+3k+2} + x_{n+3k}). \end{aligned}$$

This implies, $H^{k+1} = (1+Y_{k+1})H^0 + Y_{k+1}H^{0'} + Z_{k+1}$, where Y_{k+1} and Z_{k+1} are $k+1$ and $k+2$ degree polynomials respectively. □

Now we present the lower and upper bound on algebraic immunity of H^i in terms of the algebraic immunity of H^0.

Theorem 4. $AI_n(H_0) \leq AI_{n+3i}(H_i) \leq AI_n(H_0) + i + 2$.

Proof. To show $AI_n(H_0) \leq AI_{n+3i}(H_i)$, it is enough to show $AI_{n+3}(H^1) \geq AI_n(H^0)$. We have $H^1 = (1+x_{n+3})*F^1+x_{n+3}*G^1$ where $F^1 = x_{n+2}+x_{n+1}+H^0$ and $G^1 = x_{n+2}+x_n+H^{0'}$. Let $AI_n(H^0) = d$. So, $AI_n(H^{0'}) = d$. Following Proposition 2[case (ii)] we have $AI_{n+2}(F^1) \geq d$, and following Proposition 2[case (iii)] we have $AI_{n+2}(G^1) \geq d-1$. Then following Proposition 1, we have $AI_n(H^1) \geq d$.

Now we prove the upper bound. Following Proposition 3, we get $H^i = (1 + Y_i)H^0 + Y_iH^{0'} + Z_i$, where Y_i and Z_i are degree i and degree $i+1$ polynomials respectively. Let algebraic immunity of H^0 be d. Let there be a polynomial g^0 having degree d such that $H^0 * g^0 = 0$ or $(1 + H^0) * g^0 = 0$. Let $H^0 = p + q * x_n$ where p, q are functions on $n-1$ variables, free from the variable x_n. So, $(1 + Y_i)H^0 + Y_iH^{0'} = (1 + Y_i) * (p + q * x_n) + Y_i * (p + q * (x_{n+1} + x_{n+2})) = Y_i * q * (x_n + x_{n+1} + x_{n+2}) + p + q * x_n = Y_i * q * (x_n + x_{n+1} + x_{n+2}) + H^0$.

Construct a function $U = g^0 * (1 + Z_i) * (1 + x_n + x_{n+1} + x_{n+2})$ of degree at most $d+i+2$. Now, if $H^0 * g^0 = 0$ then $H^i * U = ((1+Y_i)H^0 + Y_iH^{0'} + Z_i) * U = (Y_i * q * (x_n + x_{n+1} + x_{n+2}) + H^0 + Z_i) * g^0 * (1 + Z_i) * (1 + x_n + x_{n+1} + x_{n+2}) = 0$. Similarly for $(1 + H^0) * g^0 = 0$, it can be shown that $(1 + H^i) * U = 0$. □

During each iteration, the algebraic immunity increases at most by 2. This is because, $H^{i+1} = (1 + x_{n+3i+3})(H^i + x_{n+3i+2} + x_{n+3i+1}) + x_{n+3i+3}(H^{i'} + x_{n+3i+2} + x_{n+3i})$. If $g, h \in B_n$ and $\deg(g), \deg(h) \leq d$ such that $H^i * g = h$ then $H^{i+1} * (1+x_{n+3i+3}) * g = (1+x_{n+3i+3})(h+g*(x_{n+3i+2}+x_{n+3i+1}))$, which shows algebraic immunity can not increase by more than two during each iteration. On the other hand, if we go for i many iterations, then the maximum increase in algebraic immunity is $i + 2$.

Example 1. Let us start with an initial $(5, 1, 3, 12)$ function $H^0 = x_5(x_1x_4 + x_3x_4 + x_2x_4 + x_2 + x_3) + x_1x_4 + x_3x_4 + x_2 + x_1$. We found the algebraic immunity of H^0, H^1, H^2, H^3 are $2, 4, 4, 5$ respectively. The function H^1 is an $(8, 3, 4, 112)$ function with $AI_8(H^1) = 4$. This function is optimized considering order of resiliency, nonlinearity, algebraic degree and algebraic immunity together. The function H^2 is an $(11, 5, 5, 992)$ function. Since the algebraic degree of this function is 5, we cannot have $AI_{11}(H^2)$ as high as $\lceil \frac{11}{2} \rceil = 6$, we can get the value 5 at maximum. We checked that the value is actually $AI_{11}(H^2) = 4$. The function H^3 is a $(14, 7, 6, 2^{13} - 2^8)$ function. Since the algebraic degree of this function is 6, we cannot have $AI_{14}(H^3)$ as high as $\frac{14}{2} = 7$, we can get the value 6 at maximum. We checked that the value is actually $AI_{14}(H^3) = 5$.

The original Maiorana-McFarland class of bent function is as follows [3]. Consider n-variable Boolean functions on (x, y), where $x, y \in \{0,1\}^{\frac{n}{2}}$ of the form $f(x, y) = x \cdot \pi(y) + g(y)$ where π is a permutation on $\{0,1\}^{\frac{n}{2}}$ and g is any Boolean function on $\frac{n}{2}$ variables. The function f can be seen as concatenation of $2^{\frac{n}{2}}$ distinct (upto complementation) affine function on $\frac{n}{2}$ variables. Similar kind of concatenation technique has also been used for construction of resilient functions [18, 16].

One idea in this direction is to concatenate k-variable affine functions (repetition may be allowed) non degenerate on at least $m+1$ variables to generate an m-resilient function f on n-variables. For such a function f, it is easy to find an annihilator g of degree $n-k+1$ as described in [13]. It has been commented in [13–Example 1 and the following paragraph] that k is generally greater than $\frac{n}{2}$ (this may be true for the Maiorana-McFarland type of functions presented in [15], but may not be true for some large class of Maiorana-McFarland type of functions described in [16, 2]) and hence it is possible to get an annihilator g of degree less than $\frac{n}{2}$. However, it should be noted that in construction of resilient functions, there are lot of techniques [16] that use concatenation of k-variable affine functions where $k < \frac{n}{2}$. In such a case, the annihilators described in [13–Theorem 2] will be of degree greater than $\frac{n}{2}$ and will not be of practical use as there are other annihilators of degree $\leq \frac{n}{2}$ which are not of the form given in [13–Theorem 2].

As example, the function H^0 in Example 1 above can be seen as concatenation of 3-variable affine functions $x_1+x_2, x_2+x_3, x_1+x_3, x_1+x_2+x_3$ non degenerate on at least two variables. In a similar fashion, the functions H^1, H^2, H^3 can also be seen as concatenation of only these four linear functions on 3-variables. Thus, it is clear that the assumption in the paper [13] that $k > \frac{n}{2}$ is not a valid assumption for $n \geq 8$ in this example.

We also like to present some interesting observations on $(9, 1, 7, 240)$ functions constructed in [16–Theorem 10(b)]. These functions can be seen as concatenation of affine functions on 3-variables, non degenerate on at least one variable. To explain this construction we briefly present some notations from [16].

Take a bit b and a bit string $s = s_0 \dots s_{n-1}$. Then the string b AND $s = s'_0 \dots s'_{n-1}$, where $s'_i = b$ AND s_i. Take two bit strings $x = x_0 \dots x_{n-1}$ and $y = y_0 \dots y_{m-1}$. The Kronecker product $x \otimes y = (x_0 \text{ AND } y) \dots (x_{n-1} \text{ AND } y)$, which is a string of length nm. The direct sum of two bit strings x, y is $x\$y = (x \otimes y^c) \oplus (x^c \otimes y)$, where x^c, y^c are bitwise complement of x, y respectively. As an example presented in [16], if $f = 01$, and $g = 0110$, then $f\$g = 01101001$. Now we present the construction for $(2p+1, 1, 2p-1, 2^{2p}-2^p)$ function as presented in [16] for $p \geq 4$.

Let $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ be the 3-variable linear functions non degenerate on two variables (i.e., the functions $x_1+x_2, x_2+x_3, x_1+x_3, x_1+x_2+x_3$) and μ_1, μ_2, μ_3 be the 3-variable linear functions non degenerate on 1 variable (i.e., the functions x_1, x_2, x_3). Let g_i be the concatenation of the 3-variable function μ_i and its complement μ_i^c, for $1 \leq i \leq 3$. That is g_i's are basically 4-variable functions. Let h_1, h_2 be bent functions on $2p-4$ variables, and h_3, h_4, h_5 be bent functions of $2p-6$ variables and h_6, h_7 be two strings of lengths $2^{2p-6}+1$ and $2^{2p-6}-1$ which are prepared by properly adding and removing 1 bit from the truth table of $(2p-6)$-variable bent functions respectively. Let f be a concatenation of the following sequence of functions. $h_1\$\lambda_1, h_2\$\lambda_2, h_3\$g_1, h_4\$g_2, h_5\$g_3, h_6\$\lambda_3, h_7\$\lambda_4$.

Example 2. For $p = 4$, we choose the functions: $h_1 = 0000010100110110, h_2 = 0000010100110110, h_3 = 0001, h_4 = 0001, h_5 = 0001, h_6 = 00010, h_7 = 001$. In this case, we find a $(9, 1, 7, 240)$ function f_1 with $AI_9(f_1) = 3$. If we change

$h_2 = 0000010100110110$ by $h_2 = 0000010100111001$, then we get a $(9, 1, 7, 240)$ function f_2 with $AI_9(f_2) = 4$.

Based on the above discussion we like to make the following comments.

(1) There are Maiorana-McFarland type of constructions (concatenation of affine functions) where the concatenation of affine functions on small number of variables is exploited. In such a case, the annihilators presented in [13] will be not of much use. Thus in line of comments presented in [9], we too argue here that there is no reason to consider that the Maiorana-McFarland type constructions are inherently weak in terms of algebraic immunity.

(2) In Example 2, we note that changing the order of affine functions can change the algebraic immunity without any change in order of resiliency, nonlinearity and algebraic degree. The change in last four bits in h_2 implies that the concatenation of $\lambda_2, 1 + \lambda_2, 1 + \lambda_2, \lambda_2$ will be replaced by $1 + \lambda_2, \lambda_2, \lambda_2, 1 + \lambda_2$. This increases the algebraic immunity from 3 to 4. It will be of great interest to study the functions presented in [16, 17, 2].

Acknowledgment. The authors like to thank the anonymous reviewers for their excellent comments that improved both the technical and editorial quality of this paper.

References

1. A. Canteaut and M. Trabbia. Improved fast correlation attacks using parity-check equations of weight 4 and 5. In *EUROCRYPT 2000*, number 1807 in Lecture Notes in Computer Science, pages 573–588. Springer Verlag, 2000.
2. C. Carlet. A larger class of cryptographic Boolean functions via a study of the Maiorana-McFarland construction. In *Advances in Cryptology - CRYPTO 2002*, number 2442 in Lecture Notes in Computer Science, pages 549–564. Springer Verlag, 2002.
3. C. Carlet. Recent results on binary bent functions. In *Proceedings of the International Conference on Combinatorics, Information Theory and Statistics*, Journal of Combinatorics, Information and System Sciences, Vol. 25, Nos. 1-4, pp. 133-149, 2000.
4. N. Courtois and J. Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - ASIACRYPT 2002*, number 2501 in Lecture Notes in Computer Science, pages 267–287. Springer Verlag, 2002.
5. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - EUROCRYPT 2003*, number 2656 in Lecture Notes in Computer Science, pages 345–359. Springer Verlag, 2003.
6. N. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003*, number 2729 in Lecture Notes in Computer Science, pages 176–194. Springer Verlag, 2003.
7. C. Ding, G. Xiao, and W. Shan. *The Stability Theory of Stream Ciphers*. Number 561 in Lecture Notes in Computer Science. Springer-Verlag, 1991.
8. E. Filiol and C. Fontaine. Highly nonlinear balanced Boolean functions with a good correlation-immunity. In *Advances in Cryptology - EUROCRYPT'98*. Springer-Verlag, 1998.

9. K. C. Gupta and P. Sarkar. Efficient software implementation of resilient Maiorana-McFarland S-Boxes. In *5th International Workshop on Information Security Applications, WISA 2004*, to be published in Lecture Notes in Computer Science. Springer-Verlag.
10. M. Hell, A. Maximov and S. Maitra. On efficient implementation of search strategy for rotation symmetric Boolean functions. In *Ninth International Workshop on Algebraic and Combinatoral Coding Theory, ACCT 2004*, June 19–25, 2004, Black Sea Coast, Bulgaria.
11. T. Johansson and F. Jonsson. Fast correlation attacks through reconstruction of linear polynomials. In *Advances in Cryptology - CRYPTO 2000*, number 1880 in Lecture Notes in Computer Science, pages 300–315. Springer Verlag, 2000.
12. A. Maximov, M. Hell and S. Maitra. Plateaued Rotation Symmetric Boolean Functions on Odd Number of Variables. IACR eprint server, eprint.iacr.org, no. 2004/144, 25 June 2004.
13. W. Meier, E. Pasalic and C. Carlet. Algebraic attacks and decomposition of Boolean functions. In *Advances in Cryptology - EUROCRYPT 2004*, number 3027 in Lecture Notes in Computer Science, pages 474–491. Springer Verlag, 2004.
14. E. Pasalic, S. Maitra, T. Johansson and P. Sarkar. New constructions of resilient and correlation immune Boolean functions achieving upper bounds on nonlinearity. In *Workshop on Coding and Cryptography - WCC 2001*, Paris, January 8–12, 2001. Electronic Notes in Discrete Mathematics, Volume 6, Elsevier Science, 2001.
15. E. Pasalic. Degree optimized resilient Boolean functions from Maiorana-McFarland class. In *9-th IMA conference on Cryptography and Coding*, 2003.
16. P. Sarkar and S. Maitra. Construction of nonlinear Boolean functions with important cryptographic properties. In *Advances in Cryptology - EUROCRYPT 2000*, number 1807 in Lecture Notes in Computer Science, pages 485–506. Springer Verlag, May 2000.
17. P. Sarkar and S. Maitra. Nonlinearity bounds and construction of resilient Boolean functions. In *Advances in Cryptology - Crypto 2000*, number 1880 in Lecture Notes in Computer Science, pages 515–532, Springer-Verlag, 2000.
18. J. Seberry, X. M. Zhang, and Y. Zheng. On constructions and nonlinearity of correlation immune Boolean functions. In *Advances in Cryptology - EUROCRYPT'93*, pages 181–199. Springer-Verlag, 1994.
19. T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, September 1984.
20. P. Stănică and S. Maitra. Rotation Symmetric Boolean Functions – Count and Cryptographic Properties. In *R. C. Bose Centenary Symposium on Discrete Mathematics and Applications*, December 2002. Electronic Notes in Discrete Mathematics, Elsevier, Volume 15.
21. P. Stănică, S. Maitra and J. Clark. Results on Rotation Symmetric Bent and Correlation Immune Boolean Functions. In *Fast Software Encryption 2004*, volume 3017 in Lecture Notes in Computer Science, pages 161–177, Springer-Verlag, 2004.
22. Y. V. Tarannikov. On resilient Boolean functions with maximum possible nonlinearity. In *Progress in Cryptology - INDOCRYPT 2000*, number 1977 in Lecture Notes in Computer Science, pages 19–30. Springer Verlag, 2000.
23. D. Wagner. A generalized birthday problem. In *Advances in Cryptology - CRYPTO 2002*, number 2442 in Lecture Notes in Computer Science, pages 288–303. Springer Verlag, 2002.

Generalized Boolean Bent Functions

Laurent Poinsot and Sami Harari

Institut des Sciences de l'Ingénieur de Toulon et du Var (I.S.I.T.V.),
Université du Sud, Toulon-Var (U.S.T.V.),
Laboratoire S.I.S., Avenue G. Pompidou, BP 56,
83162 La Valette du Var cédex, France
{laurent.poinsot, sami.harari}@univ-tln.fr

Abstract. The notions of perfect nonlinearity and bent functions are closely dependent on the action of the group of translations over $\mathbb{F}_2^m$. Extending the idea to more generalized groups of involutions without fixed points gives a larger framework to the previous notions. In this paper we largely develop this concept to define G-perfect nonlinearity and G-bent functions, where G is an Abelian group of involutions, and to show their equivalence as in the classical case.

1 Introduction

The security of secret-key cryptosystems is essentially based on the resistance to two famous attacks, *differential* [1] and *linear cryptanalysis* [2].

On the one hand the functions that exhibit the best resistance to differential cryptanalysis, called *perfect nonlinear*, satisfy to the following conditions

$$\forall \alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}, \forall \beta \in \mathbb{F}_2^n, |\{x \in \mathbb{F}_2^m | f(x \oplus \alpha) \oplus f(x) = \beta\}| = 2^{m-n} \qquad (1)$$

where $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$ and $\oplus$ is the sum over $\mathbb{F}_2^m$ and $\mathbb{F}_2^n$ (the component-wise modulo-two sum). Then for all $\alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}$, the derivative of f in the direction α, $d_\alpha f : x \in \mathbb{F}_2^m \mapsto f(x \oplus \alpha) \oplus f(x)$, is uniformly distributed over $\mathbb{F}_2^n$. On the other hand the linear resistant functions, called *bent functions*, are defined with respect to their (discrete) Fourier transform,

$$\forall \beta \in \mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}, \forall \alpha \in \mathbb{F}_2^m, \widehat{\chi_{\mathbb{F}_2^n}^{\beta} \circ f}(\alpha) = \pm 2^{\frac{m}{2}} \qquad (2)$$

where $\chi_{\mathbb{F}_2^n}^{\beta} : y \in \mathbb{F}_2^n \mapsto (-1)^{\beta.y} \in \{\pm 1\}$ is a character, the symbol "." denotes the (canonical) dot-product over $\mathbb{F}_2^n$, $\widehat{F}$ is the Fourier transform of a function $F : \mathbb{F}_2^m \longrightarrow \mathbb{C}$ and $\circ$ is the composition of functions.

Actually these two notions are equivalent as pointed out by Nyberg in [3] since

a function is perfect nonlinear if and only if it is bent.

The two corresponding attacks are dual one from the other by the Fourier transform.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 107–119, 2004.

Now let σ_α be the translation by α over $\mathbb{F}_2^m$. We can naturally rewrite the formula (1)

$$\forall \alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}, \forall \beta \in \mathbb{F}_2^n, |\{x \in \mathbb{F}_2^m | f(\sigma_\alpha(x)) \oplus f(x) = \beta\}| = 2^{m-n} \quad . \quad (3)$$

Thus the concept of perfect nonlinearity is closely linked with the action of the translations over $\mathbb{F}_2^m$.

There is a natural way to extend at the same time the notions of perfect nonlinearity and, by duality, that of bent functions. Suppose G is an Abelian group of involutions without fixed points of $\mathbb{F}_2^m$, then we can introduce the notion of *G-perfect nonlinearity* of f by considering the action of G over $\mathbb{F}_2^m$ as follows

$$\forall \sigma \in G \setminus \{Id\}, \forall \beta \in \mathbb{F}_2^n, |\{x \in \mathbb{F}_2^m | f(\sigma(x)) \oplus f(x) = \beta\}| = 2^{m-n} \quad (4)$$

where Id is the identity function of $\mathbb{F}_2^m$.

1.1 Our Contributions

In this paper we extend the notion of perfect nonlinearity by using involutions instead of simple translations. We also establish a dual version of G-perfect nonlinearity, as in the classical case, in terms of Fourier transform, that allows us to generalize the notion of bent functions. We exhibit some relations between the original and new concepts. In order to summarize we offer a larger framework to the concepts of perfect nonlinearity and bent functions.

1.2 Organization of the Paper

The continuation of this paper is organized as follows. In the next section, we give the basic definitions from dual groups to Abelian groups of involutions that are used along the paper. In Sect. 3, we introduce our new notion of G-perfect nonlinearity based on involutions. Then we study its duality through the Fourier transform in order to extend the concept of boolean bent functions. In addition, a construction of a generalized bent function is proposed. The Sect. 4 is devoted to the links between classical and new notions. Finally in Sect. 5, we show as in the classical case that our perfect nonlinear functions reach the maximum distance to a certain kind of affine functions.

2 Notations and Preliminaries

In this part we recall some essential concepts and results on dual groups, Fourier transform and bent functions. We also introduce several properties of involutions without fixed points.

2.1 Dual Group, (Discrete) Fourier Transform and Bent Functions

The definitions and results of this paragraph come from [4] and [5].

Let G be a finite Abelian group. We denote by e_G its neutral element and by E its exponent *i.e.* the maximum order of its elements. A *character* of G is any homomorphism from G to the multiplicative group of E^{th} roots of unity. The set of all characters $\widehat{G}$ is an Abelian group, called the *dual group of* G, isomorphic to G. We fix some isomorphism from G to $\widehat{G}$ and we denote by χ_G^α the image of $\alpha \in G$ by this isomorphism. Then $\chi_G^{e_G}$ is the trivial character *i.e.* $\chi_G^{e_G}(x) = 1 \forall x \in G$. For instance if $G = \mathbb{F}_2^m$, $\chi_{\mathbb{F}_2^m}^\alpha : x \in \mathbb{F}_2^m \mapsto (-1)^{\alpha.x}$. Until the end of this paper, any time we refer to a finite Abelian group, we suppose that an isomorphism from it to its dual group has been fixed.

The *Fourier transform* of any complex-valued function f on G is defined by

$$\widehat{f}(\alpha) = \sum_{x \in G} f(x)\chi_G^\alpha(x) \text{ for } \alpha \in G \quad .$$

We have the following and important lemma for the Fourier transform.

Lemma 1. *Let* $f : G \longrightarrow \mathbb{C}$.

1. $f(x) = 0$ *for every* $x \neq e_G$ *in* G *if and only if* $\widehat{f}$ *is constant.*
2. $\widehat{f}(\alpha) = 0$ *for every* $\alpha \neq e_G$ *in* G *if and only if* f *is constant.*

Let us introduce some notions needed to define the concept of bent functions. Let G_1 and G_2 be two finite Abelian groups. Let $f : G_1 \longrightarrow G_2$. f is said *balanced* if $\forall \beta \in G_2$, $|\{x \in G_1 | f(x) = \beta\}| = \frac{|G_1|}{|G_2|}$.

The *derivative of* f *in direction* $\alpha \in G_1$ is defined by

$$d_\alpha f : x \in G_1 \mapsto f(\alpha + x) - f(x) \in G_2 \tag{5}$$

where "$+$" is the symbol for the law of G_1 and "$y - z$" is an abbreviation for "$y * z^{-1}$" with $(y, z) \in G_2^2$, $*$ the law of G_2 and z^{-1} the inverse of z in G_2.
The function f is said *perfect nonlinear* if

$$\forall \alpha \in G_1 \setminus \{e_{G_1}\}, \forall \beta \in G_2, |\{x \in G_1 | d_\alpha f(x) = \beta\}| = \frac{|G_1|}{|G_2|} \quad . \tag{6}$$

Then f is perfect nonlinear if and only if for all $\alpha \in G_1 \setminus \{e_{G_1}\}$, $d_\alpha f$ is balanced.

Proposition 1. *Let* f *be any function from* G_1 *to* G_2. *Then* f *is balanced if and only if, for every* $\beta \in G_2 \setminus \{e_{G_2}\}$, *we have*

$$\widehat{\chi_{G_2}^\beta \circ f}(e_{G_1}) = 0 \quad . \tag{7}$$

We can recall the notion of bent functions : f is *bent* if $\forall \alpha \in G_1$, $\forall \beta \in G_2 \setminus \{e_{G_2}\}$, $|\widehat{\chi_{G_2}^\beta \circ f}(\alpha)| = \sqrt{|G_1|}$ where $|z|$ is the norm for $z \in \mathbb{C}$.
Finally we have the following theorem due to Nyberg.

Theorem 1. *$f : G_1 \longrightarrow G_2$ is perfect nonlinear if and only if it is bent.*

In this paper we refer to these notions as *original*, *classical* or *traditional*, as it has been already done, so as to differentiate them from ours which are qualified as *new*, *extended* or *generalized*.

2.2 Involutions Without Fixed Points

Let $S(\mathbb{F}_2^m)$ be the symmetric group of $\mathbb{F}_2^m$. Let $\sigma \in S(\mathbb{F}_2^m)$. σ is an *involution* if $\sigma^2 = \sigma \circ \sigma = Id$ or in other terms $\sigma^{-1} = \sigma$. Moreover σ is *without fixed points* if $\forall x \in \mathbb{F}_2^m$, $\sigma x \neq x$. We denote by $Inv(\mathbb{F}_2^m)$ the set of *involutions without fixed points*. By definition, we can easily see that an element of $Inv(\mathbb{F}_2^m)$ is the product of 2^{m-1} transpositions with disjoint supports. So $Inv(\mathbb{F}_2^m)$ is a conjugacy class of $S(\mathbb{F}_2^m)$. Its cardinality is given by the formula $\frac{2^m!}{2^{m-1}!2^{2^{m-1}}}$.

Let $T(\mathbb{F}_2^m)$ be the (Abelian) group of translations of $\mathbb{F}_2^m$ (subgroup of $S(\mathbb{F}_2^m)$). Then we can easily check that $T(\mathbb{F}_2^m) \setminus \{Id\} \subset Inv(\mathbb{F}_2^m)$ and since for $m > 2$, $|Inv(\mathbb{F}_2^m)| > |T(\mathbb{F}_2^m)| = 2^m$ there exists (lots of) nonlinear involutions without fixed points.

In the sequel we adopt the following usual notations, for $(\sigma, \tau) \in S(\mathbb{F}_2^m)^2$, $\sigma\tau$ and σx denote respectively $\sigma \circ \tau$ and $\sigma(x)$. The small Greek letters are kept to name the permutations and we use the small Roman letters to denote the points of $\mathbb{F}_2^m$.

We have these interesting and useful properties concerning involutions without fixed points.

Property 1. Let G be a subgroup of $S(\mathbb{F}_2^m)$ such that $G \setminus \{Id\} \subset Inv(\mathbb{F}_2^m)$ (such a group is called a *group of involutions of* $\mathbb{F}_2^m$). Then G is Abelian.

Proof. Let $(\sigma, \tau) \in G^2$. Since $\sigma\tau \in G$ then either $\sigma\tau = Id$ or $\sigma\tau \in Inv(\mathbb{F}_2^m)$. In the first case, $\sigma = \tau^{-1} = \tau$ then $\sigma\tau = \tau\sigma$. In the second case, $(\sigma\tau)^2 = Id \Leftrightarrow \sigma\tau\sigma\tau = Id \Leftrightarrow \tau\sigma\tau = \sigma^{-1} = \sigma \Leftrightarrow \sigma\tau = \tau^{-1}\sigma = \tau\sigma$.
The property follows. □

Property 2. Let G be a group of involutions of $\mathbb{F}_2^m$. Then $|G| \leq 2^m$.

Proof. Suppose on the contrary that $|G| > 2^m$. Then there exists $(\sigma, \tau) \in G^2$ such that $\sigma \neq \tau$ and $\sigma 0_{\mathbb{F}_2^m} = \tau 0_{\mathbb{F}_2^m}$. If not then $f_{0_{\mathbb{F}_2^m}} : \sigma \in G \mapsto f_{0_{\mathbb{F}_2^m}}(\sigma) = \sigma 0_{\mathbb{F}_2^m} \in \mathbb{F}_2^m$ is injective and $|\{f_{0_{\mathbb{F}_2^m}}(\sigma) | \sigma \in G\}| = |G| \leq |\mathbb{F}_2^m| = 2^m$ which is impossible by hypothesis. So let $(\sigma, \tau) \in G^2$ such that $\sigma \neq \tau$ and $\sigma 0_{\mathbb{F}_2^m} = \tau 0_{\mathbb{F}_2^m}$. Then $\sigma\tau 0_{\mathbb{F}_2^m} = \sigma\sigma 0_{\mathbb{F}_2^m} = \sigma^2 0_{\mathbb{F}_2^m} = 0_{\mathbb{F}_2^m}$. Consequently $0_{\mathbb{F}_2^m}$ is a fixed point for $\sigma\tau$. Since $\sigma \neq \tau$ then $\sigma\tau \neq Id$ and $\sigma\tau$ has no fixed point. Thus we have a contradiction with the assumption that $|G| > 2^m$. □

Property 3. For $m > 2$, there exists G a group of involutions of $\mathbb{F}_2^m$ such that $|G| = 2^m$ and $G \neq T(\mathbb{F}_2^m)$.

Proof. Let $\alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}$ and $\sigma_\alpha \in T(\mathbb{F}_2^m)$ the corresponding translation. Let $\tau \in Inv(\mathbb{F}_2^m) \setminus T(\mathbb{F}_2^m)$ (such a nonlinear involution exists since $m > 2$).

Since τ and σ_α are conjugate, there exists $\pi \in S(\mathbb{F}_2^m)$ such that $\tau = \pi\sigma_\alpha\pi^{-1}$. It is easy to see that $\pi T(\mathbb{F}_2^m)\pi^{-1}$ is a group of involutions (a *conjugate group of* $T(\mathbb{F}_2^m)$) such that $|\pi T(\mathbb{F}_2^m)\pi^{-1}| = 2^m$ and $\pi T(\mathbb{F}_2^m)\pi^{-1} \neq T(\mathbb{F}_2^m)$ (since $\tau \in \pi T(\mathbb{F}_2^m)\pi^{-1}$ and $\tau \notin T(\mathbb{F}_2^m)$). □

Remark 1. In the previous property, the fact "$m > 2$" is needed to obtain a group of involutions G such that $|G| = 2^m$ **and** $G \neq T(\mathbb{F}_2^m)$. If $m = 1$ or $m = 2$ we have only one group of involutions of maximal size $G = T(\mathbb{F}_2)$ or $G = T(\mathbb{F}_2^2)$.

We call *maximal group of involutions* of $\mathbb{F}_2^m$ a group of involutions G of $\mathbb{F}_2^m$ such that $|G| = 2^m$.

Property 4. Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Then the action $\phi : G \longrightarrow S(\mathbb{F}_2^m)$ such that $\phi(\sigma) : x \mapsto \sigma x$ is simply transitive.

Proof. Let us define for $x \in \mathbb{F}_2^m$ the orbital function $f_x : \sigma \in G \mapsto f_x(\sigma) = \phi(\sigma)(x) = \sigma x \in \mathbb{F}_2^m$. Then for all $x \in \mathbb{F}_2^m$, f_x is injective. Indeed let $(\sigma, \tau) \in G^2$ such that $\sigma \neq \tau$. If $f_x(\sigma) = f_x(\tau)$ then we have the following chain of equivalences $\sigma x = \tau x \Leftrightarrow \tau\sigma x = x \Leftrightarrow x$ is a fixed point of $\tau\sigma$ which is impossible since $\tau\sigma \neq Id$. In addition we have $|G| = |\mathbb{F}_2^m|$ then f_x is bijective. That concludes the proof. □

Finally for a group of involutions G of $\mathbb{F}_2^m$, since the exponent of G is 2 (all the elements distinct from the identity have an order two) and it is an Abelian group, the dual group $\widehat{G}$ is the set of homomorphisms from G to $\{\pm 1\}$ and is isomorphic to G.

3 Generalized Boolean Bent Functions

In this section we introduce a new notion of perfect nonlinearity that extends and offers a larger framework for the classical one. We also study its dual version through the Fourier transform which leads us to introduce a generalized definition for the concept of bent functions.

3.1 Definitions and Properties

Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$. We define the *derivative of* f *in direction* $\sigma \in G$ by

$$\begin{aligned} D_\sigma f : \mathbb{F}_2^m &\longrightarrow \mathbb{F}_2^n \\ x &\mapsto D_\sigma f(x) = f(\sigma x) \oplus f(x) \ . \end{aligned} \tag{8}$$

We define $\Delta_f = \sup_{\sigma \neq Id, \beta} |\{x \in \mathbb{F}_2^m | D_\sigma f(x) = \beta\}|$.

We have the following bound for Δ_f.

Theorem 2. *For any function* $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$, $\Delta_f \geq 2^{m-n}$.

Proof. For each fixed $\sigma \in G \setminus \{Id\}$, the collection of sets $\{\{x \in \mathbb{F}_2^m | D_\sigma f(x) = \beta\}\}_{\beta \in \mathbb{F}_2^n}$ is a partition of $\mathbb{F}_2^m$. Then $\sum_{\beta \in \mathbb{F}_2^n} |\{x \in \mathbb{F}_2^m | D_\sigma f(x) = \beta\}| = 2^m$, which implies the result. □

Definition 1. *A function $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$ is G-perfect nonlinear if $\Delta_f = 2^{m-n}$.*

According to the previous theorem, for a G-perfect nonlinear function f, we have

$$\Delta_f = \inf_{g:\mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n} \Delta_g \ . \tag{9}$$

We can state a first result similar to the traditional case.

Theorem 3. *$f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$ is G-perfect nonlinear if and only if for all $\sigma \in G \setminus \{Id\}$, the derivative $D_\sigma f$ is balanced.*

Proof. f is G-perfect nonlinear if and only if the maximum of the sequence of integers $\{|\{x \in \mathbb{F}_2^m | D_\sigma f(x) = \beta\}|\}_{\sigma \in G \setminus \{Id\}, \beta \in \mathbb{F}_2^n}$ is equal to its mean. This is possible if and only if the sequence is constant. Then the constant must be 2^{m-n} which ensures the result. □

From the theorem above we obtain the following immediate results which embeds classical notions in our framework.

Proposition 2. *Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$. f is $T(\mathbb{F}_2^m)$-perfect nonlinear if and only if f is perfect nonlinear in the classical way.*

Proof. f is $T(\mathbb{F}_2^m)$-perfect nonlinear if and only if $D_{\sigma_\alpha} f$ is balanced for every $\sigma_\alpha \in T(\mathbb{F}_2^m) \setminus \{Id\}$ if and only if $D_{\sigma_\alpha} f$ is balanced for every $\alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}$. We conclude the proof since $D_{\sigma_\alpha} f(x) = d_\alpha f(x)$ for all $x \in \mathbb{F}_2^m$. □

We now develop the dual description of G-perfect nonlinear functions through the study of their Fourier transform.
Let f and g be two functions from $\mathbb{F}_2^m$ to $\mathbb{R}$. We define

$$\begin{aligned} \Phi_{f,g} : G &\longrightarrow \mathbb{R} \\ \sigma &\mapsto \Phi_{f,g}(\sigma) = \sum_{x \in \mathbb{F}_2^m} f(x) g(\sigma x) \end{aligned} \tag{10}$$

which can be seen as a kind of convolution product with respect to the action of G over $\mathbb{F}_2^m$. Let us compute its Fourier transform. Let $\sigma \in G$.

$$\begin{aligned} \widehat{\Phi_{f,g}}(\sigma) &= \sum_{\tau \in G} \Phi_{f,g}(\tau) \chi_G^\sigma(\tau) \\ &= \sum_{\tau \in G} \sum_{x \in \mathbb{F}_2^m} f(x) g(\tau x) \chi_G^\sigma(\tau) \\ &= \sum_{x \in \mathbb{F}_2^m} f(x) \sum_{\tau \in G} g(\tau x) \chi_G^\sigma(\tau) \ . \end{aligned} \tag{11}$$

Moreover the sum "$\sum_{\tau\in G} g(\tau x)\chi_G^\sigma(\tau)$" is invariant by translations [1] over G *i.e.* $\forall \pi \in G,\ \sum_{\tau\in G} g(\tau x)\chi_G^\sigma(\tau) = \sum_{\tau\in G} g(\tau\pi x)\chi_G^\sigma(\tau\pi) = \sum_{\tau\in G} g(\tau\pi x)\chi_G^\sigma(\tau)\chi_G^\sigma(\pi)$. Then we have

$$\begin{aligned}(11) &= \sum_{x\in\mathbb{F}_2^m} f(x)\chi_G^\sigma(\pi)\sum_{\tau\in G} g(\tau\pi x)\chi_G^\sigma(\tau)\\ &= \sum_{x\in\mathbb{F}_2^m} f(\pi x)\chi_G^\sigma(\pi)\sum_{\tau\in G} g(\tau x)\chi_G^\sigma(\tau)\ (\textit{since } \pi^{-1}=\pi)\\ &= \sum_{x\in\mathbb{F}_2^m} f(\pi x)\chi_G^\sigma(\pi)\widehat{g_x}(\sigma) \qquad (12)\end{aligned}$$

where $g_x : G \longrightarrow \mathbb{R}$ such that $g_x(\sigma) = g(\sigma x)$. Since (12) is true for all $\pi \in G$, by integration over G, we obtain

$$\begin{aligned}\sum_{\pi\in G}\widehat{\Phi_{f,g}}(\sigma) &= |G|\widehat{\Phi_{f,g}}(\sigma) = 2^m\widehat{\Phi_{f,g}}(\sigma)\\ &= \sum_{x\in\mathbb{F}_2^m}\sum_{\pi\in G} f(\pi x)\chi_G^\sigma(\pi)\widehat{g_x}(\sigma)\\ &= \sum_{x\in\mathbb{F}_2^m}\widehat{f_x}(\sigma)\widehat{g_x}(\sigma)\ . \qquad (13)\end{aligned}$$

And finally this gives us

$$\forall\sigma\in G,\ \widehat{\Phi_{f,g}}(\sigma) = \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^m}\widehat{f_x}(\sigma)\widehat{g_x}(\sigma) \qquad (14)$$

which is equivalent, in our context, to the trivialization of the convolution product by the Fourier transform.

Proposition 3. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$, $\beta \in \mathbb{F}_2^n$ and $F_{\beta,f} : G \longrightarrow \mathbb{R}$ such that $F_{\beta,f}(\sigma) = \widehat{\chi^\beta_{\mathbb{F}_2^n} \circ D_\sigma f(0_{\mathbb{F}_2^m})}$. Then we have*

$$\forall\sigma\in G,\ \widehat{F_{\beta,f}}(\sigma) = \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^m}(\widehat{\chi^\beta_{\mathbb{F}_2^n}\circ f_x}(\sigma))^2.$$

Proof. First of all, $F_{\beta,f}$ is real-valued since the characters of $\mathbb{F}_2^m$ and $\mathbb{F}_2^n$ are $\{\pm1\}$-valued.

[1] $\tau \in G \mapsto \tau\pi \in G$ is the translation by $\pi \in G$.

Let us compute the Fourier transform of $F_{\beta,f}$.

$$\begin{aligned}
\widehat{F_{\beta,f}}(\sigma) &= \sum_{\tau\in G} F_{\beta,f}(\tau)\chi_G^\sigma(\tau)\\
&= \sum_{\tau\in G}\sum_{x\in\mathbb{F}_2^m} (\chi^\beta_{\mathbb{F}_2^n}\circ D_\tau f)(x)\chi_G^\sigma(\tau)\\
&= \sum_{\tau\in G}\sum_{x\in\mathbb{F}_2^m} \chi^\beta_{\mathbb{F}_2^n}(f(x)\oplus f(\tau x))\chi_G^\sigma(\tau)\\
&= \sum_{\tau\in G}\sum_{x\in\mathbb{F}_2^m} (\chi^\beta_{\mathbb{F}_2^n}\circ f)(x)(\chi^\beta_{\mathbb{F}_2^n}\circ f)(\tau x)\chi_G^\sigma(\tau)\\
&= \sum_{\tau\in G} \Phi_{\chi^\beta_{\mathbb{F}_2^n}\circ f,\chi^\beta_{\mathbb{F}_2^n}\circ f}(\tau)\chi_G^\sigma(\tau)\\
&= \widehat{\Phi_{\chi^\beta_{\mathbb{F}_2^n}\circ f,\chi^\beta_{\mathbb{F}_2^n}\circ f}}(\sigma)\\
&= \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^m} \widehat{(\chi^\beta_{\mathbb{F}_2^n}\circ f)}_x(\sigma)\widehat{(\chi^\beta_{\mathbb{F}_2^n}\circ f)}_x(\sigma) \text{ (according to (14))}\\
&= \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^m} (\widehat{(\chi^\beta_{\mathbb{F}_2^n}\circ f)}_x(\sigma))^2\\
&= \frac{1}{2^m}\sum_{x\in\mathbb{F}_2^m} \widehat{(\chi^\beta_{\mathbb{F}_2^n}\circ f_x}(\sigma))^2 \ .
\end{aligned}$$

□

Then we have one of the most important theorem which allows us to define an extended notion of bent functions.

Theorem 4. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f:\mathbb{F}_2^m\longrightarrow\mathbb{F}_2^n$. f is G-perfect nonlinear if and only if $\forall\sigma\in G$, $\forall\beta\in\mathbb{F}_2^n\setminus\{0_{\mathbb{F}_2^n}\}$,*

$$\sum_{x\in\mathbb{F}_2^m} \widehat{(\chi^\beta_{\mathbb{F}_2^n}\circ f_x}(\sigma))^2 = 2^{2m} \ .$$

Proof. f is G-perfect non linear $\Leftrightarrow \forall\sigma\in G\setminus\{Id\}$, $D_\sigma f$ is balanced over $\mathbb{F}_2^m$
$\Leftrightarrow \forall\sigma\in G\setminus\{Id\}$, $\forall\beta\in\mathbb{F}_2^n\setminus\{0_{\mathbb{F}_2^n}\}$, $\widehat{\chi^\beta_{\mathbb{F}_2^n}\circ D_\sigma f}(0_{\mathbb{F}_2^m})=0$ (by *proposition* 1)
$\Leftrightarrow \forall\beta\in\mathbb{F}_2^n\setminus\{0_{\mathbb{F}_2^n}\}$, $\forall\sigma\in G\setminus\{Id\}$, $F_{\beta,f}(\sigma)=0$
$\Leftrightarrow \forall\beta\in\mathbb{F}_2^n\setminus\{0_{\mathbb{F}_2^n}\}$, $\widehat{F_{\beta,f}}$ is constant over G (according to *lemma* 1).

By Parseval equation we have $\frac{1}{2^m}\sum_{\sigma\in G}(\widehat{F_{\beta,f}}(\sigma))^2 = \sum_{\sigma\in G}(F_{\beta,f}(\sigma))^2 = (F_{\beta,f}(Id))^2$. Thus since $\widehat{F_{\beta,f}}$ is constant, $(\widehat{F_{\beta,f}}(\sigma))^2 = (F_{\beta,f}(Id))^2$ for all $\sigma\in G$. Moreover $F_{\beta,f}(Id) = \widehat{\chi^\beta_{\mathbb{F}_2^n}\circ D_{Id}f}(0_{\mathbb{F}_2^m}) = \sum_{x\in\mathbb{F}_2^m}\chi^\beta_{\mathbb{F}_2^m}(0_{\mathbb{F}_2^m}) = 2^m$. Then according to *proposition* 3 we deduce the result. □

We can then define the new boolean bent functions by the duality through the Fourier transform previously exhibited as follows.

Definition 2. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$. f is called G-bent if* $\forall \sigma \in G$, $\forall \beta \in \mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}$, $\sum_{x \in \mathbb{F}_2^m} (\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x(\sigma)})^2 = 2^{2m}$.

By this way we keep the equivalence (by *theorem* 4) between the new notions of perfect nonlinearity and bent functions as it is the case for the original concepts.

3.2 Construction of a G-Perfect Nonlinear Function

Let $\pi \in S(\mathbb{F}_2^m)$ and $G_\pi = \pi T(\mathbb{F}_2^m)\pi^{-1}$ the conjugate group of $T(\mathbb{F}_2^m)$ by π (it is a maximal group of involutions). Suppose that there exists $g : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$ such that g is perfect nonlinear in the classical way (so g is also bent in the classical way). Let define $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$ by $f(x) = g(\pi^{-1}x)$. We have then the following proposition.

Proposition 4. *The function f previously defined is G_π-perfect nonlinear.*

Proof. Let $\sigma \in G_\pi \setminus \{Id\}$ and $\beta \in \mathbb{F}_2^n$. We have

$$|\{x \in \mathbb{F}_2^m | f(\sigma x) \oplus f(x) = \beta\}| = |\{x \in \mathbb{F}_2^m | f(\pi\sigma_\alpha\pi^{-1}x) \oplus f(x) = \beta\}| \quad (15)$$

since there exists one and only one $\alpha \in \mathbb{F}_2^m \setminus \{0_{\mathbb{F}_2^m}\}$ such that the translation σ_α is conjugated by π with σ. Then we have

$$\begin{aligned}(15) &= |\{y \in \mathbb{F}_2^m | f(\pi\sigma_\alpha y) \oplus f(\pi y) = \beta\}| \text{ (change of variable : } y = \pi^{-1}x) \\ &= |\{y \in \mathbb{F}_2^m | g(\sigma_\alpha y) \oplus g(y) = \beta\}| \\ &= 2^{m-n} \text{ (by perfect nonlinearity of g) .}\end{aligned}$$

That concludes the proof. □

4 Links Between Classical and New Notions

In this section we present some relations between our new notions and the classical ones.

Theorem 5. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$.*
f is G-perfect nonlinear if and only if $\forall x \in \mathbb{F}_2^m$, $f_x : G \longrightarrow \mathbb{F}_2^n$ such that $f_x(\sigma) = f(\sigma x)$ is perfect nonlinear in the traditional sense.

Proof.

⇒) Suppose that f is G-perfect nonlinear. We have to prove that $\forall x \in \mathbb{F}_2^m$, $\forall \sigma \in G \setminus \{Id\}$ and $\forall \beta \in \mathbb{F}_2^n$,

$$|\{\tau \in G | f_x(\sigma\tau) \oplus f_x(\tau) = \beta\}| = \tfrac{|G|}{2^n} = 2^{m-n}.$$

We have $|\{\tau \in G | f(\sigma\tau x) \oplus f(\tau x) = \beta\}| = |\{y \in \mathbb{F}_2^m | f(\sigma y) \oplus f(y) = \beta\}|$ by the change of variables $\tau x = y$ (one must remember that $\tau \mapsto \tau x$ is bijective since the action of G on $\mathbb{F}_2^m$ is simply transitive). That concludes the first implication.

$\Leftarrow$) Suppose that $\forall x \in \mathbb{F}_2^m$, f_x is perfect nonlinear in the classical way. Then f_x is bent (also in the original sense) *i.e.* $\forall \beta \in \mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}$, $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = \sqrt{|G|} = 2^{\frac{m}{2}}$. Then we have $\sum_{x \in \mathbb{F}_2^m} (\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma))^2 = \sum_{x \in \mathbb{F}_2^m} |G| = 2^{2m}$. So f is G-perfect nonlinear by *theorem* 4. □

Then we have the immediate following corollary, similar to the traditional case.

Corollary 1. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$.*
f is G-perfect nonlinear if and only if $\forall x \in \mathbb{F}_2^m$, $\forall \beta \in \mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}$, $\forall \sigma \in G$, $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = 2^{\frac{m}{2}}$.

5 Distance to "Affine" Functions

A well-known result is that bent functions have the maximum distance to the set of affine functions defined by the canonical dot-product. In this section we show a similar result. The bent functions with respect to the extended notion reach the maximum distance between a certain kind of affine functions as in the classical context.

Let f and b be functions from E_1 to E_2 (two sets and E_1 is finite), we define the Hamming distance between f and g by

$$d(f, g) = |\{x \in E_1 | f(x) \neq g(x)\}| \ . \tag{16}$$

If A is a (finite) set of functions from E_1 to E_2 we define the distance of a function $f : E_1 \longrightarrow E_2$ to the set A by

$$d(f, A) = \min_{g \in A} d(f, g) \ . \tag{17}$$

Let G be a maximal group of involutions of $\mathbb{F}_2^m$. We define the set of "*affine functions*" over G as

$$\begin{aligned} \mathcal{A}_G &= \{f : G \longrightarrow \{\pm 1\} | \exists (\lambda, c) \in \widehat{G} \times \{\pm 1\} \text{ such that } f(\sigma) = c\lambda(\sigma)\} \\ &= \{\pm \chi^{\sigma}_G | \sigma \in G\} \end{aligned}$$

i.e. $\mathcal{A}_G$ is the set of affine forms over G.

Let $(\beta, x) \in (\mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}) \times \mathbb{F}_2^m$. We have

$$\begin{aligned}\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma) &= \sum_{\tau \in G} \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x))\chi^{\sigma}_G(\tau) \\ &= |\{\tau \in G | \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x)) = \chi^{\sigma}_G(\tau)\}| - |\{\tau \in G | \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x)) \neq \chi^{\sigma}_G(\tau)\}| \\ &\quad \text{(since both } \chi^{\beta}_{\mathbb{F}_2^n} \text{ and } \chi^{\sigma}_G \text{ are } \{\pm 1\} - \text{valued)} \\ &= |G| - 2d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \chi^{\sigma}_G) \ .\end{aligned}$$

Thus we obtain

$$d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \chi^{\sigma}_G) = 2^{m-1} - \frac{1}{2}\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma) \ . \tag{18}$$

Let us compute $d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, -\chi^{\sigma}_G)$.

$$\begin{aligned}d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, -\chi^{\sigma}_G) &= |\{\tau \in G | \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x)) \neq -\chi^{\sigma}_G(\tau)\}| \\ &= |\{\tau \in G | \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x)) = \chi^{\sigma}_G(\tau)\}| \\ &= |G| - |\{\tau \in G | \chi^{\beta}_{\mathbb{F}_2^n}(f(\tau x)) \neq \chi^{\sigma}_G(\tau)\}| \\ &= |G| - d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \chi^{\sigma}_G) \\ &= 2^{m-1} + \frac{1}{2}\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma) \ .\end{aligned}$$

It follows that $d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \{\pm\chi^{\sigma}_G\}) = \min(\{d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \chi^{\sigma}_G), d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, -\chi^{\sigma}_G)\})$ $= 2^{m-1} - \frac{1}{2}|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)|$.

Since $\mathcal{A}_G = \cup_{\sigma \in G}\{\pm\chi^{\sigma}_G\}$, we have

$$\begin{aligned}d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \mathcal{A}_G) &= \min_{\alpha \in \mathcal{A}_G} d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \alpha) \\ &= \min_{\sigma \in G} d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \{\pm\chi^{\sigma}_G\}) \\ &= \min_{\sigma \in G}(2^{m-1} - \frac{1}{2}|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)|) \\ &= 2^{m-1} - \frac{1}{2}\max_{\sigma \in G}|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| \ .\end{aligned} \tag{19}$$

Proposition 5. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$ and $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$.*
f is G-perfect nonlinear if and only if $\forall(\beta, x) \in (\mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}) \times \mathbb{F}_2^m$,

$$d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \mathcal{A}_G) = 2^{m-1} - 2^{\frac{m}{2}-1} \ .$$

Proof.

$\Leftarrow$) Let $g : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$. By Parseval equation, we have $\sum_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x}(\sigma)|^2 = |G| \sum_{\sigma\in G} |\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x(\sigma)|^2 = |G|^2$ (since $\chi^{\beta}_{\mathbb{F}_2^n}$ is $\{\pm 1\}$-valued). So $\max_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x}(\sigma)| \geq \sqrt{|G|} = 2^{\frac{m}{2}}$ and then $\inf_{g:\mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n} \max_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x}(\sigma)| \geq 2^{\frac{m}{2}}$. Moreover suppose that we have $d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \mathcal{A}_G) = 2^{m-1} - 2^{\frac{m}{2}-1}$, then according to formula (19), we deduce that $\forall(\beta, x) \in (\mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}) \times \mathbb{F}_2^m$, $\max_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = 2^{\frac{m}{2}}$. Then $\forall\sigma \in G$, $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| \leq 2^{\frac{m}{2}}$. The lower absolute bound previously exhibited implies then that $\forall\sigma \in G$, $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = 2^{\frac{m}{2}}$. The result is given by *corollary* 1.

$\Rightarrow$) By *corollary* 1, if f is G-perfect nonlinear then $\forall(\beta, x) \in (\mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}) \times \mathbb{F}_2^m$, $\forall\sigma \in G$, $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = 2^{\frac{m}{2}}$. Therefore we deduce the result by applying the formula (19). □

Corollary 2. *Let G be a maximal group of involutions of $\mathbb{F}_2^m$. Let $f : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$. If f is G-perfect nonlinear then $\forall(\beta, x) \in (\mathbb{F}_2^n \setminus \{0_{\mathbb{F}_2^n}\}) \times \mathbb{F}_2^m$, $\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x$ has the maximal distance to $\mathcal{A}_G$.*

Proof. Suppose f G-perfect nonlinear. The same way as in the proof of *proposition* 5, we deduce the $|\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x}(\sigma)| = \inf_{g:\mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n} \max_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x}(\sigma)| = 2^{\frac{m}{2}}$. Then $\forall g : \mathbb{F}_2^m \longrightarrow \mathbb{F}_2^n$, according to formula (19),

$$d(\chi^{\beta}_{\mathbb{F}_2^n} \circ f_x, \mathcal{A}_G) \geq 2^{m-1} - \frac{1}{2} \max_{\sigma\in G} |\widehat{\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x}(\sigma)| = d(\chi^{\beta}_{\mathbb{F}_2^n} \circ g_x, \mathcal{A}_G) \ .$$

□

6 Conclusion and Further Works

We have extended both notions of perfect nonlinearity and bent functions, while respecting the equivalence between them, by considering groups of involutions rather than the simple translations. Moreover we have shown that our concepts and the original ones are closely dependent. Finally we have obtained a similar result to the traditional case with regard to the distance to the set of affine functions.

The existence of G-perfect nonlinear functions is proved by our construction of such function in the case where G is a conjugate group of the group of translations $T(\mathbb{F}_2^m)$. A problem remaining to solve is to show if the conjugacy class of $T(\mathbb{F}_2^m)$ is equal to the set of all maximal groups of involutions of $\mathbb{F}_2^m$. If it is not the case, we should also construct a G-bent function for G a group of involutions which is not in the same conjugacy class than $T(\mathbb{F}_2^m)$, or we should prove their nonexistence.

References

[1] E. Biham, A. Shamir : Differential Cryptanalysis of DES-like Cryptosystems. *Journal of Cryptology*, Vol. 4, No. 1, pp. 3-72, 1991

[2] M. Matsui : Linear Cryptanalysis Method for DES Cipher. *Advances in Cryptology, Proc. Eurocrypt'93, LNCS 809*, pp. 1-17, 1994

[3] K. Nyberg : Perfect nonlinear S-boxes. In *Lecture Notes in Computer Science, Advances in Cryptology - EUROCRYPT'91*, volume 547, pp. 378-385. Springer-Verlag, 1991

[4] C. Carlet, C. Ding : Highly nonlinear mappings. In *Journal of Complexity*, Volume 20, Issue 2-3, Special issue on Coding and Cryptography, pp. 205-244, 2004

[5] O.A. Logachev, A.A. Salnikov, V.V. Yashchenko : Bent functions on a finite Abelian group, *Discrete Math. Appl.* 7(6), pp. 547-564, 1997

On Boolean Functions with Generalized Cryptographic Properties*

An Braeken[1,**], Ventzislav Nikov[2], Svetla Nikova[1,***], and Bart Preneel[1]

[1] Department Electrical Engineering, ESAT/COSIC,
Katholieke Universiteit Leuven, Kasteelpark Arenberg 10,
B-3001 Heverlee-Leuven, Belgium
{an.braeken, svetla.nikova}@kuleuven.ac.be
[2] Department of Mathematics and Computing Science,
Eindhoven University of Technology,
P.O. Box 513, 5600 MB, Eindhoven, the Netherlands
v.nikov@tue.nl

Abstract. By considering a new metric, we generalize cryptographic properties of Boolean functions such as resiliency and propagation characteristics. These new definitions result in a better understanding of the properties of Boolean functions and provide a better insight in the space defined by this metric. This approach leads to the construction of "handmade" Boolean functions, i.e., functions for which the security with respect to some specific monotone sets of inputs is considered, instead of the security with respect to all possible monotone sets with the same cardinality, as in the usual definitions. This approach has the advantage that some trade-offs between important properties of Boolean functions can be relaxed.

1 Introduction

For any two binary vectors $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$ in $\mathbb{F}_2^n$, define the sets $\delta(x, y) = \{i : x_i \neq y_i\}$ and $\sup(x) = \{i : x_i \neq 0\}$. Denote the size of a set A with $|A|$. Then the Hamming distance between the binary vectors x and y is equal to $d(x, y) = |\delta(x, y)|$ and the Hamming weight of x is $\mathrm{wt}(x) = |\sup(x)|$. It was noted that $\delta(x, y)$ has properties similar to metric and $\sup(x)$ has properties similar to norm [NN03].

Our goal is to use $\delta(x, y)$ instead of the Hamming distance and $\sup(x)$ instead of the Hamming weight and to explore the properties of this new space. For this

* The work described in this paper has been supported in part by the European Commission through the IST Programme under Contract IST-2002-507932 ECRYPT and Concerted Research Action GOA-MEFISTO-666 of the Flemish Government.

** The author is research assistant of the Fund for Scientific research - Flanders (Belgium).

*** The author was partially supported by IWT STWW project on Anonymity and Privacy in Electronic Services.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 120–135, 2004.

purpose we consider monotone increasing and monotone decreasing sets. A set Δ is called monotone decreasing if for each set in Δ, its subsets belong to Δ. Similarly, a set Γ is said to be monotone increasing if for each set in Γ its supersets belong to Γ.

As it has already been shown in [NN03], this new space with monotone sets can be used to generalize notions such as codes, minimum distance of a code, minimal codewords, generator and parity check matrices of a code, packing and covering, error-correcting capabilities, etc. In addition, monotone sets are widely used in Secret Sharing Schemes (SSS) to describe the sets of players which are allowed (disallowed) to reconstruct a secret. It has been recently pointed out [NN03] that the security of (verifiable) SSS can be derived from the properties of this space.

This paper focuses on Boolean functions. In particular, we generalize the definition of *t-resilient* functions to functions which are resilient with respect to a monotone decreasing set Δ. Analogously, the parameters for defining the *propagation characteristics* (PC) of functions are replaced by monotone decreasing sets. Our aim is to provide a new insight to the previous results and to give a better understanding of which structural properties contribute in which way to known results.

Very often the properties of resiliency and PC imply strong requirements to the rest of the parameters of a Boolean function. This leads to some trade-offs between them, since all relevant properties cannot be satisfied simultaneously. For example, Siegenthaler's inequality [S84] states that $d \leq n - t - 1$, where d is the algebraic degree, n is the dimension and t is the order of resiliency. By exactly defining which components need to satisfy a certain order of resiliency or PC, we can strengthen the weaker components by using other constructions and achieve in this way an optimal design.

By means of example, we present a modified version of the combination generator (see Section 3.5 for concrete examples). Let Δ be the set consisting of all subsets of LFSRs for which the sum of the lengths is shorter than the security parameter for the (fast) correlation attack [S85, MS92, JJ99]. It is known that the feedback polynomials of the combining LFSRs should be primitive with distinct degrees, not necessary co-prime, in order to obtain maximum linear complexity [RS87]. Using t-resilient functions the degrees of LFSRs' polynomials are uniformly chosen. But considering Δ-resilient functions instead, allows us to choose the degrees non-uniformly as well as to relax the requirements to the rest of the function parameters like nonlinearity, algebraic degree, etc. Using a Δ-resilient function as combiner f, the (fast) correlation attack can be avoided. Moreover, the degree of the function f should be high in order to counter the linear synthesis by Berlekamp-Massey [M69] and algebraic attacks [CM03]. Note that in this model the trade-off defined by the Siegenthaler's inequality can be relaxed to another form as shown in Section 3.2.

In order to preclude more recent algebraic attacks, we should also require that the function has no low degree multiples [CM03]. To get even better security, but a small trade-off in speed, one can replace some linear feedback shift registers

by nonlinear feedback shift registers or clock controlled linear feedback shift registers, since the algebraic attacks of [CM03] do not apply on this model. The set Δ for defining the resiliency contains again the subsets of LFSRs for which the sum of the lengths is smaller than the security parameter for the (fast) correlation attack.

The first steps in considering generalizations of classical t-resiliency and functions satisfying PC properties has been made in [CCCF00]. The authors extended the properties of resiliency and propagation characteristics with respect to subspaces. So, our definitions can be seen as natural extensions of the definitions by Canteaut et al., instead of subspaces, to collections of subspaces.

We also refer to the research on almost resilient functions and functions satisfying almost PC properties [KJS01, K99, DSS01]. There, the concept is different and is based on probabilities but it is also introduced for relaxing the parameters and for avoiding (or relaxing) the trade-offs.

The paper is organized as follows. In Section 2, we give some background and preliminaries. Section 3 deals with Δ-resilient functions. We first investigate the notions of algebraic degree, nonlinearity and divisibility results for the Walsh coefficients. Then different constructions are identified amongst the other we mention the constructions of Siegenthaler, Camion et al., Maiorana-MacFarland, and the Direct sum constructions. We also give two concrete example of Δ-resilient functions that have better trade-off between degree/nonlinearity and resiliency compared with the classical theory. In Section 4 we generalize functions which satisfy SAC and PC for some monotone decreasing sets. Then a relation between them and Δ-resilient functions is proven. In this setting we also investigate the question when a function may possess linear structures. Finally we investigate the algebraic degree and show a generalization of Kurosawa and Satoh's construction of PC functions using a relation between monotone span programs and linear codes. Most of the proofs are skipped due to the page limit.

2 Background

Define the set $\mathcal{P} = \{1, \ldots, n\}$ and denote the power set of $\mathcal{P}$ by $P(\mathcal{P})$. The set Γ ($\Gamma \subseteq P(\mathcal{P})$) is called *monotone increasing* if for each set A in Γ, each set containing A is also in Γ. Similarly, the set Δ ($\Delta \subseteq P(\mathcal{P})$) is called *monotone decreasing*, if for each set B in Δ each subset of B is in Δ. A monotone increasing set Γ can be described efficiently by the set Γ^- consisting of the minimal elements (sets) in Γ, i.e., the elements in Γ for which no proper subset is also in Γ. Similarly, the set Δ^+ consists of the maximal elements (sets) in Δ, i.e., the elements in Δ for which no proper superset is also in Δ. We set $\Gamma = \Delta^c$ ($\Delta^c = P(\mathcal{P}) \setminus \Delta$). Note that Γ is monotone increasing if and only if Δ is monotone decreasing.

The *dual* sets $\Delta^\perp$ and $\Gamma^\perp$ are defined by $\Gamma^\perp = \{A : A^c \in \Delta\}$ and $\Delta^\perp = \{A : A^c \in \Gamma\}$. It is easy to see that $\Delta^\perp$ is monotone decreasing and $\Gamma^\perp$ is monotone increasing. For two monotone decreasing sets Δ_1 and Δ_2 let us define

$\Delta_1 \uplus \Delta_2 = \{A = A_1 \cup A_2; A_1 \in \Delta_1, A_2 \in \Delta_2\}$. Note that $\Delta_1 \uplus \Delta_2$ is again a monotone decreasing set.

As it has been pointed out in [NN03], $\delta(x, y)$ satisfies the properties of a metric and $\sup(x)$ the properties of a norm. Notice that $\sup(x)$ and $\delta(x, y) = \sup(x-y)$ are subsets of $\mathcal{P}$ and that $\mathcal{P}$ is partially ordered (i.e., $x \preceq y$ if and only if $\sup(x) \subseteq \sup(y)$). For a vector $u \in \mathbb{F}_2^n$, let $\overline{u} = u \oplus 1$ (where 1 denotes the all-1 vector), i.e., $\sup(\overline{u}) = \sup(u)^c$. The dot product $w \cdot x$ is equal to the componentwise inner product.

For an element $A \in \Delta \setminus \{0\}$, the subspace defined by A is given by $U_A = \{u : \sup(u) \subseteq A\}$. The dual $U_A^\perp$ of the subspace U_A is the subspace consisting of the elements x such that $x \cdot y = 0$ for all $y \in U_A$. Consequently, $U_A^\perp$ is defined by A^c, i.e., $U_A^\perp = U_{A^c} = \{u : \sup(u) \subseteq A^c\}$.

Let $f(x) = f(x_1, \ldots, x_n)$ be a Boolean function on $\mathbb{F}_2^n$. The *Walsh transform* W_f of a Boolean function $f(x)$ plays an important role in our work. It is a real-valued function, which is defined as follows $W_f(w) = \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+w \cdot x}$. A function with equally distributed outputs is called a balanced function. It is clear that for balanced functions $W_f(0) = 0$. A Boolean function $f(x)$ on $\mathbb{F}_2^n$ is said to be a *plateaued* function [CaPr03, ZZ99b] if its Walsh transform W_f takes only 0 and $\pm\lambda$, where λ is a positive integer, called the *amplitude* of the plateaued function.

The *nonlinearity* N_f of a Boolean function f is defined by the minimum distance of the function to the set of affine functions $\mathcal{A}$, i.e., $N_f = \min_{g \in \mathcal{A}} d(f, g)$, or also $N_f = 2^{n-1} - \frac{1}{2} \max_{w \in \mathbb{F}_2^n} |W_f(w)|$.

Another representation of a Boolean function $f(x)$ is the *algebraic normal form* (ANF) $f(x) = \bigoplus_{u \in \mathbb{F}_2^n} a_u x^u, \ \ a_u \in \mathbb{F}_2$. The degree of the ANF is called the *algebraic degree* or shortly *degree* (denoted by $\deg(f)$) of the Boolean function. The *autocorrelation* r_f of a Boolean function f on $\mathbb{F}_2^n$ is a real-valued transformation, defined by $r_f(u) = 2^{-n} \sum_{x \in \mathbb{F}_2^n} (-1)^{f(x)+f(x+u)}$. We will also need an important property of the sum of characters (see [J92–p. 263]).

Lemma 1. *For any subspace $V \subseteq \mathbb{F}_2^n$, we have*

$$\sum_{x \in V} (-1)^{w \cdot x} = \begin{cases} |V| & \text{if } w \in V^\perp; \\ 0 & \text{otherwise.} \end{cases}$$

3 Δ-Resilient Functions

3.1 Definition and Relation with the Classical Definition of Resiliency

In this section we generalize the definitions of resilient and correlation-immune (CI) functions with respect to a monotone decreasing set Δ. We assume that the set Δ is the maximal possible monotone decreasing set for which the function satisfies the corresponding property. The monotone increasing set Γ corresponding to Δ is defined by $\Gamma = \Delta^c$.

Definition 1. *Let $f(x) = f(x_1, \ldots, x_n)$ be a Boolean function on $\mathbb{F}_2^n$ and Δ be a monotone decreasing set. Then $f(x)$ is called* **Δ-resilient** *if and only if $f(x) \oplus w \cdot x$ is a balanced function for all w such that $\sup(w) \in \Delta$. Furthermore, $f(x)$ is called* **Δ-CI** *if and only if $f(x) \oplus w \cdot x$ is a balanced function for all w such that $\sup(w) \in \Delta \setminus \{\emptyset\}$.*

When $\Delta = \{A : |A| \leq t\}$ the definitions of Δ-resilient function and t-resilient function, (resp. Δ-CI function and t-CI function) coincide. The balancedness property of $f(x) \oplus w \cdot x$ can be translated in terms of Walsh spectrum into $W_f(w) = 0$. Denote the set of vectors which have zero Walsh value by ZW_f, then $\Delta \subseteq \{sup(u) : u \in \mathrm{ZW}_f\}$. Note that $\mathrm{ZW}_f \cap \Gamma$ is not necessarily empty.

Example 1. Consider the sets Δ^+ and Γ^- in the set $\mathbb{F}_2^4$: $\Delta^+ = \{\{1,2\},\{3,4\}\}$ and $\Gamma^- = \{\{1,4\},\{2,4\},\{1,3\},\{2,3\}\}$. It is easy to verify that $\Gamma = \Delta^c$ and $\Gamma \cap \Delta = \emptyset$. A function which is Δ-resilient has zero Walsh coefficients for the inputs w, where $\sup(w) \in \{\emptyset,\{1\},\{2\},\{3\},\{4\},\{1,2\},\{3,4\}\}$, i.e., for the vectors $w \in \{(0,0,0,0),(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(1,1,0,0),(0,0,1,1)\}$.

Next we establish the relationship with the classical definition of resiliency. For the monotone sets Γ and Δ define the parameters

$$t_1 = \min\{|A| : A \in \Gamma^-\} \qquad \text{and} \qquad t_2 = \max\{|A| : A \in \Delta^+\}.$$

From the definition of t_1 and the fact that Γ is a monotone increasing set, each subset of size $t_1 - 1$ belongs to Δ, which implies that a Δ-resilient function is also $(t_1 - 1)$-resilient. Analogously, a Δ-CI function is $(t_1 - 1)$-CI. The parameter t_2 defines the maximum dimension of a subspace in which the Δ-resilient function is resilient.

The following theorem shows a necessary and sufficient condition for Δ-resilient functions concerning their balancedness properties on affine subspaces.

Theorem 1. *A Boolean function f on $\mathbb{F}_2^n$ is Δ-resilient if and only if f is balanced when restricted to any of the affine subspaces $a + U_A$, where $A \in \Delta^\perp$.*

Remark 1. From the definition of resiliency, we deduce that if at most t components of a t-resilient function are fixed (this defines a subspace V of dimension $n - t$), the output is balanced. The previous theorem generalizes this property by proving that the function is also balanced on all affine subspaces of $V^\perp$.

Example 2. A possible truth table of the Δ-resilient function defined by Example 1 is given by the vector $(0,1,0,1,1,1,0,0,0,0,1,1,1,0,1,0)$. This function is exactly 1-resilient. Moreover the function is resilient with respect to two subspaces of dimension 2 whose bases are given by $< e_1, e_2 >$ and $< e_3, e_4 >$, where e_i is the all zero vector except for position i. One can check that the conditions of Theorem 1 are satisfied.

3.2 Algebraic and Numerical Degree

Theorem 2. *For a Δ-resilient function f on $\mathbb{F}_2^n$ all ANF coefficients a_u of f with $\sup(u) \in \Gamma^\perp$ and $wt(u) > 1$ are equal to zero. If $\sup(u) \in \Gamma^\perp$ and $wt(u) = 1$ then $a_u = 1$.*

Proof. The Siegenthaler's inequality $\deg(f) \leq n - t - 1$ for t-resilient functions on $\mathbb{F}_2^n$ relies on the observation that the coefficient a_u of the term x^u in the ANF of f satisfies the following relation [XM88]

$$a_u = 2^{wt(u)-1} - 2^{-wt(\overline{u})-1} \sum_{w \preceq \overline{u}} W_f(w) \mod 2\,. \tag{1}$$

Consider now u with $\sup(u) \in \Gamma^\perp$: then $\sup(\overline{u}) \in \Delta$ and $\sup(w) \subseteq \sup(\overline{u}) \in \Delta$ for all $w \preceq \overline{u}$. By definition of Δ-resilient functions $W_f(w) = 0$ for $\sup(w) \in \Delta$. Therefore $a_u = 0$ for all u such that $\sup(u) \in \Gamma^\perp$ and $wt(u) > 1$, but when $\sup(u) \in \Gamma^\perp$ and $wt(u) = 1$ we obtain $a_u = 1$. Note that this is a generalization of the Siegenthaler's inequality for t-resilient functions: if $\Delta = \{A : |A| \leq t\}$ we have $\Gamma^\perp = \{B : |B| \geq n - t\}$. □

Remark 2. Notice that because of the factor mod 2 in (1) the coefficient a_u is 1 for any u such that $\sup(u) \subseteq [\Delta^\perp]^+$ and $W_f(\overline{u}) = \pm 2^{n-wt(u)+1}$. The maximum weight of such a u defines the normal algebraic degree of the Boolean function. Knowledge of the coefficients of the ANF of f enables us to derive (upper and lower) bounds on the nonlinearity as shown in [ZZI99–Theorem 18 and 30].

We now generalize the definition of degree to this new setting.

Definition 2. *Define a monotone decreasing set $Deg = \{A : A \subseteq \sup(u), a_u \neq 0\}$. We call the set Deg^+ the "degree-set" of f.*

Remark 3. The "degree-set" of f satisfies the following relation $Deg \subseteq \Delta^\perp \cup \{A : A \in \Gamma^\perp, |A| = 1\}$. Moreover, the equality does not always hold; it is even possible that $Deg^+ \cap [\Delta^\perp]^+ = \emptyset$.

Example 3. Applying Theorem 2 to the function of Example 1, we obtain that all coefficients a_u for u such that $\sup(u) \in \Gamma^\perp$ are zero, which gives additional information compared to the Siegenthaler's inequality. Note that $[\Gamma^\perp]^- = \{\{3,4\},\{1,2\}\}$ and $[\Delta^\perp]^+ = \{\{2,4\},\{2,3\},\{1,4\},\{1,3\}\}$. Because the ANF of f is given by $x_1x_3 \oplus x_1x_4 \oplus x_2x_3 \oplus x_2x_4 \oplus x_1 \oplus x_3$, the equality $Deg^+ = [\Delta^\perp]^+$ holds in this example.

Theorem 3. *For a Δ-resilient function $f(x)$ on $\mathbb{F}_2^n$ all coefficients λ_u from NNF of $g(x) = f(x) \oplus x_1 \oplus \cdots \oplus x_n$ with $\sup(u) \in \Gamma^\perp$ are equal to zero. Moreover, all coefficients λ_u from NNF of g with $\sup(u) \in [\Delta^\perp]^+$ are non-zero.*

3.3 Nonlinearity

In this section we improve the divisibility results on the Walsh coefficients of resilient functions which leads to an upper bound on the nonlinearity. Let f_v be the $(n - wt(v))$-variable function formed from f for which $x_j = 0$ if $v_j = 1$. The divisibility result by Sarkar and Maitra [SM00] can be generalized in the following way:

Theorem 4. *Let f be a Δ-resilient function on $\mathbb{F}_2^n$. Then the Walsh coefficients of f satisfy the following divisibility conditions:*

$$W_f(v) = 0 \mod 2^{t_3(v)+1}, \text{ where } sup(v) \in \Gamma \text{ and}$$
$$t_3(v) = \min\{wt(w) : w \preceq v, sup(w) \in \Gamma^-\}.$$

Remark 4. Note that $t_3(v) \geq t_1 = t + 1$ for v with $sup(v) \in \Gamma$, therefore we have a stronger result comparing to the divisibility of 2^{t+2} proven in [SM00] for t-resilient functions.

Now we extend the divisibility result of Carlet and Sarkar in [CS02], namely that $W_f(v) = 0 \mod 2^{t+2+\left\lfloor \frac{n-t-2}{\deg(f)} \right\rfloor}$.

Theorem 5. *Let f be a Δ-resilient function on $\mathbb{F}_2^n$. Then the Walsh coefficients of f satisfy the following divisibility conditions:*

$$W_f(v) = 0 \mod 2^{t_3(v)+1+\left\lfloor \frac{n-t_3(v)-1}{t_4(v)} \right\rfloor},$$

where $sup(v) \in \Gamma$ and with parameters $t_3(v)$ (as defined in Theorem 4) and $t_4(v) = \max\{|A| : A \in Deg^+, A \subseteq \text{sup}(\overline{u}) \text{ with } u \preceq v, sup(u) \in \Gamma^-\}$.

Proof. In [CS02], the following relation has been proven

$$\sum_{u \preceq v} W_f(u) = 2^{wt(v)} W_{f_{\overline{v}}}(0) = 2^n - 2^{wt(v)+1} wt(f_v). \tag{2}$$

Let f be a Δ-resilient function. If $sup(v) \in \Gamma^-$, then for any $u \precneqq v$ we have $u \in \Delta$ and thus $W_f(u) = 0$. Hence, Equation (2) reduces to $W_f(v) = 2^n - 2^{wt(v)+1} wt(f_v)$. Applying McEliece's [MS] theorem for cyclic codes on f_v we obtain that $wt(f_v) = 0 \mod 2^{\left\lfloor \frac{n-t_3(v)-1}{t_4(v)} \right\rfloor}$, since $t_3(v) = wt(v)$ and $t_4(v) = \deg(f_v)$. This proves the result for v with $sup(v) \in \Gamma^-$.

Let $sup(v) \in \Gamma \backslash \Gamma^-$. By the hypothesis $W_f(u) = 0 \mod 2^{t_3(u)+1+\left\lfloor \frac{n-t_3(u)-1}{t_4(u)} \right\rfloor}$ for any $u \precneqq v$ and $sup(u) \in \Gamma$. Since $t_4(u)$ is increasing with respect to $wt(u)$ we obtain that $W_f(u) = 0 \mod 2^{t_3(u)+1+\left\lfloor \frac{n-t_3(u)-1}{t_4(v)} \right\rfloor}$ for any $u \precneqq v$ and $sup(u) \in \Gamma$. Note that by Remark 3 the degree of f_v is less than or equal to $t_4(v)$. Rewrite (2) in the form $W_f(v) = 2^n - 2^{wt(v)+1} wt(f_v) - \sum_{u \precneqq v} W_f(u)$. To conclude the proof note that $t_3(u)$ is decreasing with respect to $wt(u)$ and that $t_4(v) \geq \deg(f_v)$. □

Remark 5. The parameters $t_3(v)$ and $t_4(v)$ satisfy $t_3(v) + t_4(v) \leq n$, which is similar to the Siegenthaler's inequality. Thus, Theorem 5 improves the result from Theorem 4 when $t_3(v)$ and/or $t_4(v)$ are smaller.

Example 4. Consider again the function of Example 1. By definition of Γ^- and Deg^+ (see Example 3), the parameters $t_3(v) = 2$ and $t_4(v) = 1$ for all $v \in \Gamma$. Consequently, the Walsh values of the function are divisible by 8.

The divisibility results of the Walsh coefficients for Δ-resilient functions result in bounds on the nonlinearity of these functions similar to the one of [CS02].

3.4 Constructions of Δ-Resilient Functions

Lemma 2. *If f is a Δ-resilient function on $\mathbb{F}_2^n$, then $g(x) = f(x) \oplus 1$ and $h(x) = f(x_1 \oplus c_1, \ldots, x_n \oplus c_n)$, where $c \in \mathbb{F}_2^n$ are Δ-resilient.*

The Constructions of Siegenthaler and Camion et al.

Theorem 6. *Let f_1 and f_2 be two Δ-resilient functions on $\mathbb{F}_2^n$. The function f on $\mathbb{F}_2^{n+1}$ defined by*

$$f(x_1, \ldots, x_{n+1}) = x_{n+1} f_1(x_1, \ldots, x_n) \oplus (1 \oplus x_{n+1}) f_2(x_1, \ldots, x_n)$$

is $\widetilde{\Delta}$-resilient, where $\widetilde{\Delta} = \Delta \uplus P(\{n+1\})$. Furthermore, if $w \in \Gamma$ and for any $u \preceq w$ it holds that $W_{f_1}(u) + W_{f_2}(u) = 0$ then f is $\hat{\Delta}$-resilient, where $\hat{\Delta} = \widetilde{\Delta} \cup P(\mathrm{sup}(w))$.

Remark 6. We extend Siegenthaler's result [S84] that states "if f_1 and f_2 are t-resilient then f is t-resilient" by showing that if f_1 and f_2 are Δ-resilient, then f is $\widetilde{\Delta}$-resilient. Similarly, we generalize the result of Camion et al. [CCCS92] which states "if also for all v such that $wt(v) = t + 1$ holds that $W_{f_1}(v) + W_{f_2}(v) = 0$, then f is $(t+1)$-resilient". This is here reflected in the $\hat{\Delta}$-resiliency of f.

The following construction can be seen as a special case of the previous one.

Lemma 3. *Let f_1 be a Δ-resilient function on $\mathbb{F}_2^n$. Then the functions*

$$f(x_1, \ldots, x_{n+1}) = f_1(x_1, \ldots, x_n) \oplus 0.x_{n+1}$$
$$g(x_1, \ldots, x_{n+1}) = f_1(x_1, \ldots, x_n) \oplus x_{n+1}(f_1(x_1, \ldots, x_n) \oplus f_1(x_1 \oplus 1, \ldots, x_n \oplus 1))$$

are $\Delta \uplus P(\{n+1\})$-resilient functions on $\mathbb{F}_2^{n+1}$, and the function

$$h(x_1, \ldots, x_{n+1}) = f_1(x_1, \ldots, x_n) \oplus x_{n+1}$$

is a $(\Delta \uplus P(\{n+1\})) \cup P(\{1, \ldots, n\})$-resilient function on $\mathbb{F}_2^{n+1}$.

The following corollary can be derived from Theorem 3.

Corollary 1. *Let $f(x) = w \cdot x$ be a linear function on $\mathbb{F}_2^n$ and $wt(w) = d$, i.e., without lost of generality we can suppose that $f(x) = x_1 \oplus \ldots \oplus x_d$. Then $f(x)$ is $(\cup_{i=1}^{d} P(\{1, \ldots, n\} \setminus \{i\}))$-resilient function.*

Direct Sum and Secondary Constructions

Theorem 7. *Let f_1 be a Δ_1-resilient function on $\mathbb{F}_2^{n_1}$ and f_2 be a Δ_2-resilient function on $\mathbb{F}_2^{n_2}$ then the direct sum*

$$f : \mathbb{F}_2^{n_1} \times \mathbb{F}_2^{n_2} : (x, y) \mapsto f(x, y) = f_1(x) \oplus f_2(y)$$

is a $(\widetilde{\Delta} = \Delta_1 \uplus \Delta_2 \uplus S)$-resilient function on $\mathbb{F}_2^{n_1+n_2}$ where $S = \{\emptyset, \{1\}, \cdots, \{n_1\}, \{n_1 + 1\}, \cdots, \{n_2 + n_1\}\}$.

Remark 7. The classical theorem says that for the direct sum of a t_1-resilient function and t_2-resilient function yields a $(t_1 + t_2 + 1)$-resilient function [ZZ97], which is reflected here by the set $\widetilde{\Delta}$.

The following lemma shows how to construct new Δ'-resilient functions from a given Δ-resilient function where $\Delta' \subseteq \Delta$. This theorem is an extension of Theorem 3 [C97a].

Lemma 4. *Consider a Boolean function f on $\mathbb{F}_2^n$ which is Δ-resilient. If there exists a subspace W and a subset $\Delta' \subseteq \Delta$ such that $U_A \cap W = \emptyset$ for all $A \in \Delta'$ and the restriction of f to $W^\perp$ is equal to the constant c, then the function f' obtained from f by replacing the constant c by the constant $c \oplus 1$ for all elements of $W^\perp$ is Δ'-resilient.*

The following construction is a generalization of the change of basis construction.

Lemma 5. *Let Δ be a set containing less than n elements. Then any Boolean function f on $\mathbb{F}_2^n$ which has at least n linearly independent vectors $w \in \mathbb{F}_2^n$ such that $W_f(w) = 0$ can be transformed into a Δ-resilient function.*

The Maiorana-MacFarland Construction

Theorem 8. *Let ϕ be a function from $\mathbb{F}_2^{n-r}$ into $\mathbb{F}_2^r$ and let g be an arbitrary Boolean function on $\mathbb{F}_2^{n-r}$, then the function f defined by*

$$\mathbb{F}_2^r \times \mathbb{F}_2^{n-r} \to \mathbb{F}_2 : (x, y) \mapsto f(x, y) = x \cdot \phi(y) \oplus g(y)$$

is Δ-resilient with $\Delta = \{A : \exists y \in \mathbb{F}_2^{n-r}, \text{ such that } \sup(\phi(y)) \subseteq A\}^c$. Moreover, if ϕ is injective (resp. takes each value exactly 2 times), the function is plateaued with amplitude 2^r (resp. 2^{r+1}).

Remark 8. This construction always leads to $P(\{r + 1, \ldots, n\}) \subseteq \Delta$ because ϕ is a mapping from $\mathbb{F}_2^{n-r}$ into $\mathbb{F}_2^r$. It is clear that the higher the weight of the elements in the image of ϕ are, the higher the values t_2 and $|\Delta|$ are.

3.5 Example of Modified Combination Generator

We give some concrete examples of the modified combination generator as explained in the introduction.

1. Suppose the generator consists of 5 LFSRs of lengths 61, 63, 21, 31, and 33 respectively. Let the security parameter for the (fast) correlation attack be equal to 60. Consequently in order to be secure against the (fast) correlation attack, we need a combination function which is resilient with respect to the 3^{rd}, 4^{th}, 5^{th} and also the 3^{rd}+4^{th}, 3^{rd}+5^{th} LFSR, i.e. a Δ resilient function with $\Delta = \{\{3,4\},\{3,5\}\}$. The function $f(x_1,\ldots,x_5) = x_2x_3x_4x_5 \oplus x_1x_2x_3 \oplus x_1x_4 \oplus x_3x_5 \oplus x_1 \oplus x_2$ satisfies this property. Remark that this function has degree 4 and nonlinearity 10. High degree and high nonlinearity are important properties for resisting other attacks like for instance Berlekamp-Massey attack [M69], algebraic attack [CM03] and best affine approximation attack [DXS91].
2. The function $f(x_1,\ldots,x_5) = x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_4$ is a Δ-resilient function with $\Delta = \{\{1,2\},\{1,4\},\{1,5\},\{2,4\},\{2,5\}\}$. Moreover, the function has degree 3 and maximum nonlinearity 12. The LFSRs of the corresponding modified combination generator with security parameter 60 should have for instance lengths 19, 21, 61, 31, and 33 respectively.

When we consider the same models of combination generators in the classical theory, the combination function should be in both cases 2-resilient in order to resist (fast) correlation attacks. Following Siegenthaler's inequality, the corresponding function has degree less than or equal to 2. Note that now using Δ-resilient functions the choice of the lengths of the LFSRs may not be uniform, which is the case when we use t-resilient functions. This also allows to relax the requirements to the rest of the parameters like nonlinearity, algebraic degree, etc. Moreover, by Carlet-Sarkar's result on the divisibility of the Walsh coefficients, the maximum Walsh value is greater or equal than 16, resulting in a nonlinearity less than or equal to 8.

These examples are just illustrative and need to be scaled up in order to be used in reality. However, it already shows the advantages of considering resiliency with respect to specified monotone sets since the strong trade-offs between resiliency and degree, resiliency and nonlinearity can be avoided.

4 Functions Satisfying Propagation Characteristics with Respect to Δ-Sets

Analogously to the definitions of Δ-resilient and Δ-correlation immune (CI) function, we define functions which satisfy the propagation characteristic of degree Δ_1 and of order Δ_2 (PC(Δ_1) of order Δ_2), the propagation characteristic of degree Δ_1 (PC(Δ_1)), and the strict avalanche criteria of order Δ_2 (SAC(Δ_2)), where $\Delta, \Delta_1, \Delta_2$ are monotone decreasing sets.

Definition 3. *For two monotone decreasing sets Δ_1 and Δ_2 the function f satisfies* **PC(Δ_1) of order** Δ_2 *if and only if for every w, such that* $\sup(w) \in \Delta_1 \setminus \{\emptyset\}$ *the function $f(x) \oplus f(x \oplus w)$ is Δ_2-CI. If $\Delta_2 = \emptyset$, the function f is said to be* **PC(Δ_1)**. *If $\Delta_1 = \{A : |A| = 1\}$, the function f satisfies* **SAC(Δ_2)**.

Again if $\Delta_1 = \{A : |A| \leq \ell\}$ and $\Delta_2 = \{B : |B| \leq k\}$ the definitions of PC(Δ_1) function of order Δ_2 and PC(ℓ) function of order k, PC(Δ_1) function and PC(ℓ) function; SAC(Δ_2) function and SAC(k) function coincide. The property balancedness of $f(x) \oplus f(x \oplus w)$ implies for the autocorrelation $r_f(w) = 0$.

4.1 A Relation with Δ-Resilient Functions

We generalize the well-known relation $p+t \leq n-1$ between the order of resiliency t and the degree of propagation p of a Boolean function on $\mathbb{F}_2^n$ as proven in [ZZ00, ChPa02].

Theorem 9. *For a Δ_1-resilient function on $\mathbb{F}_2^n$ which satisfies PC of degree Δ_2, we have that $\Delta_2 \cap \Gamma_1^{\perp} = \emptyset$ and $\Delta_1 \cap \Gamma_2^{\perp} = \emptyset$.*

Proof. The following relation, with respect to any linear subspace V, was derived in [CCCF01]:

$$\sum_{u \in V} r_f(u) = \frac{1}{|V^{\perp}|} \sum_{x \in V^{\perp}} W_f(x)^2 \,. \tag{3}$$

Let A be an arbitrary element of $\Delta_2 \setminus \{0\}$. Note that the coefficient $r_f(0)$ is equal to 2^n. Now applying the definition of PC of degree Δ_2 we obtain $\sum_{u \in U_A} r_f(u) = r_f(0) = 2^n = \frac{1}{|U_A^{\perp}|} \sum_{x \in U_A^{\perp}} W_f(x)^2$.

Thus $|U_A^{\perp}|\ 2^n = \sum_{x \in U_A^{\perp}} W_f(x)^2 = \sum_{x \in U_{A^c}} W_f(x)^2$. As a consequence $A^c \notin \Delta_1$ or also $A \notin \Gamma_1^{\perp}$ because otherwise the right side of the equation above would be zero. This holds for all $A \in \Delta_2$ and thus $\Delta_2 \cap \Gamma_1^{\perp} = \emptyset$, which is equivalent to $\Delta_2 \subseteq \Delta_1^{\perp}$. This in turn is equivalent to $\Gamma_2^{\perp} \subseteq \Gamma_1$, equivalent to $\Delta_1 \subseteq \Delta_2^{\perp}$ and finally equivalent to $\Delta_1 \cap \Gamma_2^{\perp} = \emptyset$. □

4.2 Linear Structures

Next we derive a condition for the existence of linear structures for a Δ_1-resilient function which satisfies $PC(\Delta_2)$. A *linear structure* of a function is an element $a \in \mathbb{F}_2^n$ for which $f(x) \oplus f(x \oplus a)$ is a constant. Linear structures should be avoided, for example, in order to resist differential attacks [B93].

Theorem 10. *Let f be a Δ_1-resilient function on $\mathbb{F}_2^n$ that satisfies $PC(\Delta_2)$. If there exists a non-empty element $A \in \Delta_2^{+} \cap [\Delta_1^{\perp}]^{+}$, then all b with* $\sup(b) = B$, *$B \in \Gamma_2^{-}$ and $A \subset B$ are linear structures of f.*

Proof. Let $A \in \Delta_2^{+} \cap [\Delta_1^{\perp}]^{+}$. From (3) for $V = U_A$ and the assumption, we deduce that there exists x, such that $\sup(x) = A^c \in \Gamma_1^{-}$ and $W_f(x)^2 = 2^n |U_A^{\perp}|$

since $W_f(y) = 0\ \forall y \in U_A^{\perp},\ y \neq x\ (\sup(y) \in \Delta_1)$. Next we apply (3) for $V = U_B$, where $B \in \Gamma_2^-$ and $A \subset B$:

$$r_f(0) + r_f(b) = \frac{2}{|U_A^{\perp}|} \sum_{x \in U_B^{\perp}} W_f(x)^2 .$$

Because $U_B^{\perp} \subseteq U_A^{\perp}$, there are two possibilities: either $\sup(x) \subseteq U_B^{\perp}$, which leads to $r_f(b) = 2^n$; or $\sup(x) \nsubseteq U_B^{\perp}$, which leads to $r_f(b) = -2^n$. The fact that $|r_f(b)| = 2^n$ implies that b is a linear structure of f. □

The following theorem gives a condition on the existence of linear structures for functions which satisfy $PC(\Delta)$. The proof is similar to the one of Theorem 10.

Theorem 11. *Let f be a Boolean function on $\mathbb{F}_2^n$ that satisfies $PC(\Delta)$. If there exists an element $x \in \mathbb{F}_2^n \setminus \{0\}$ such that $\sup(x) \in A^{\perp}$ for $A \in \Delta^+$ which satisfies $W_f(x) = 2^{n-\frac{|U_A|}{2}}$, then all b with $\sup(b) = B$ and $B \in \Gamma^-, A \subseteq B$ are linear structures of f.*

4.3 Algebraic Degree

First note that the functions satisfying $PC(\mathcal{P}(P))$ are the perfect nonlinear functions (bent functions of characteristic two). From the definition of resiliency, we deduce that for a Boolean function on $\mathbb{F}_2^n$ which satisfies $PC(\Delta_1)$ of order Δ_2, the functions $f(x) \oplus f(x \oplus w)$ are Δ_2-resilient for all $w \in \Delta_1 \setminus \{0\}$. By Theorem 1, the functions $f(x) \oplus f(x \oplus w)$ are balanced for all $w \in \Delta_1 \setminus \{0\}$ on any of the subspaces $a + U_A$, where $A \in \Delta_2^{\perp}$.

The following theorem generalizes the bound on the degree d of a function on $\mathbb{F}_2^n$ satisfying the SAC(k) property [PVV+91], namely $d \leq n - k - 1$.

Theorem 12. *If f satisfies SAC of order Δ then all coefficients a_u from the ANF of f with $\sup(u) \in \Gamma^{\perp}$ are equal to zero. Moreover, for all sets $A \in \Gamma^{\perp}$: $|A| > 1$.*

Corollary 2. *For functions satisfying $PC(\Delta_1)$ of order Δ_2, where $\forall\ A \in \Gamma_2^{\perp}$: $|A| > 1$, the ANF coefficients a_u of f with $\sup(u) \in \Gamma_2^{\perp}$ are equal to zero.*

4.4 Constructions

The set of functions which satisfy $PC(\Delta_1)$ of order Δ_2 are globally invariant under the complementation of any of its coordinates, composition with any permutation on $\{1, \ldots, n\}$ which keeps Δ_1, Δ_2 invariant, and the addition of any affine function. We first generalize the change of basis construction.

Theorem 13. *Let Δ be a set containing less than n elements. Then any Boolean function f on $\mathbb{F}_2^n$ which has at least n linearly independent vectors w such that $r_f(w) = 0$ can be transformed into a function that satisfies the PC criterion of degree Δ.*

In [NN03], many coding theoretic notions are generalized in this new setting. A generalization of the linear $[n, k, d]$-code is called an *error-set correcting* code. We slightly change the original notation here and call an $[n, k, \Delta]$-code $\widetilde{C}$ a code of dimension k, length n and for which codewords x satisfy $\sup(x) \in \Gamma$, where $\Gamma = \Delta^c$. The generator matrix of the code $\widetilde{C}$ can be defined by using the matrix M of a Monotone Span Program.

Definition 4. *[KW93] A Monotone Span Program (MSP) $\mathcal{M}$ is defined by the quadruple $(\mathbb{F}, M, \epsilon, \psi)$, where $\mathbb{F}$ is a finite field, M is a matrix (with m rows and $d \leq m$ columns) over $\mathbb{F}$, $\psi : \{1, \ldots, m\} \to \{1, \ldots, n\}$ is a surjective functions and $\epsilon = (1, 0, \ldots, 0)$ is a fixed non-zero vector, called target vector. The size of $\mathcal{M}$ is the number of rows and is denoted as size($\mathcal{M}$).*

The properties that matrix M posses are in one-to-one correspondence with a monotone increasing set Γ. It is said that M computes Γ.

Definition 5. *[NN03] An MSP is called Δ-non-redundant (denoted by Δ-rMSP) when $v \in \ker(M^T) \iff \sup(v) \in \Gamma$ ($\Gamma = \Delta^c$).*

It is shown in [NN03] how the generator matrix of an $[n, k, \Delta]$-code can be deduced from the previous results.

Theorem 14. *Let $\mathcal{M}$ be a Δ-rMSP computing Γ and let $M^\perp$ be the matrix of the dual $\mathcal{M}^\perp$ MSP computing $\Gamma^\perp$. Then a generator matrix G of an $[n, k, \Delta]$-code is given by $G = (M^\perp)^T$.*

The best known and general construction for PC(ℓ) functions of order k is due to Kurosawa and Satoh [KS97]. This construction uses linear codes. It was later generalized by Carlet [C97b] who also takes nonlinear codes into account. We present a further generalization.

Theorem 15. *Let g be an arbitrary function on $\mathbb{F}_2^s$ and Q be an $s \times t$-matrix. Define Δ_1 on $\{1, \ldots, t\}$ and Δ_2 on $\{t+1, \ldots, t+s\}$. Let M_1 be a matrix in Δ_1-rMSP computing Γ_1 and $M_1^\perp$ be the matrix in $\Delta_1^\perp$-rMSP computing $\Gamma_1^\perp$. Let M_2 be a matrix in Δ_2-rMSP computing Γ_2 and $M_2^\perp$ be the matrix in $\Delta_2^\perp$-rMSP computing $\Gamma_2^\perp$. Let $G_1 = (M_1^\perp)^T$ be the generator matrix of a $[t, h, \Delta_1]$-code and let $G_2 = (M_2^\perp)^T$ be the generator matrix of a $[s, h, \Delta_2]$-code. Define the function f on $\mathbb{F}_2^{s+t}$ as follows:*

$$f(x_1, \ldots, x_s, y_1, \ldots, y_t) = [x_1, \ldots, x_s]Q[y_1, \ldots, y_t]^T \oplus g(x_1, \ldots, x_s)\,.$$

Set $Q = G_2^T G_1$ then the function f satisfies PC(Δ_ℓ) of order Δ_k, where $\Delta_\ell = \Delta_1^\perp \uplus \Delta_2^\perp$ and $\Delta_k = \Delta_1 \uplus \Delta_2$.

Remark 9. It is easy to verify that $\Delta_k \subseteq \Delta_l^\perp$, which corresponds to $k+\ell \leq n-1$.

5 Conclusions and Open Problems

In this paper we have shown that many classical notions, constructions and results from the theory of cryptographic properties of Boolean functions can

be extended to a more general setting: t-resiliency and PC properties can be represented as Δ-resiliency or PC properties with respect to Δ, where $\Delta = \{A : |A| \leq t\}$. Instead of working with numbers, we work with sets, which give us more flexibility in satisfying incompatible requirements as shown in Section 3.5. We have also defined an analogous notion for the algebraic degree of a Boolean function. Then we have proven equivalent results to most of the known inequalities in this new setting. It is much easier to adjust the parameters of a function, when one works with sets compared to numbers. When a trade-off needs to be achieved between parameters of a function, we can easily reduce a set (e.g., Δ) with some of its elements in order to satisfy the condition, comparing to the previous case where we need to reduce the number (e.g., t to $t-1$ for example) discarding all sets of a fixed cardinality (e.g., with cardinality t).

This approach gives more insight and better understanding in the behaviour of a Boolean function. More precisely, it allows us to determine which structural properties contribute to different known results like for instance the Siegenthaler's inequality. Future work will investigate if these insights lead to new constructions of t-resilient functions (functions satisfying PC properties) by going over special monotone set resilient functions (PC functions).

We leave as an open question whether such functions exist for any Δ. In the theory of Secret Sharing Schemes (SSS), a scheme (or equivalently a monotone increasing set) is called *ideal* if each player has a share of minimal size. But it is known that for "many" monotone sets there is no ideal scheme, i.e., there is no finite field in which the SSS is ideal. For Boolean functions we consider only this ideal case, since every coordinate (input) in the function is considered as a player's share. Thus in the binary field there are monotone sets Γ for which there does not exist a corresponding MSP (equivalently SSS). We do not know a relation between MSPs and Δ-resilient functions, but it seems likely that there exist sets Δ for which there does not exist a corresponding Δ-resilient function.

Acknowledgments

The authors would like to thank Yuri Borissov and the anonymous referees for the helpful remarks and comments.

References

[B93] E. Biham, Differential Cryptanalysis of the Full 16-Round DES, *Crypto 1992, LNCS 740,* Springer-Verlag, pp. 487-496, 1993.

[CCCS92] P. Camion, C. Carlet, P. Charpin, N. Sendrier, On Correlation Immune Functions, *Crypto 1991, LNCS 576,* Springer-Verlag, pp. 86-100, 1992.

[CCCF00] A. Canteaut, C. Carlet, P. Charpin, C. Fontaine, Propagation Characteristics and Correlation-Immunity of Highly Nonlinear Boolean Functions, *Eurocrypt 2000, LNCS 1807,* Springer-Verlag, pp. 507-522, 2000.

[CCCF01] A. Canteaut, C. Carlet, P. Charpin, C. Fontaine, On Cryptographic Properties of the Cosets of $RM(1,m)$, *IEEE Trans. Information Theory,* Vol. 47(4), pp. 1494-1513, 2001.

[C97a] C. Carlet, More Correlation-Immune and Resilient Functions over Galois Fields and Galois Rings, *Eurocrypt 1997, LNCS 1233,* Springer-Verlag, pp. 422-433, 1997.

[C97b] C. Carlet, On Cryptographic Propagation Criteria for Boolean Functions, *Information and Computation,* Vol. 151(1-2), pp. 32-56, 1999.

[CaPr03] C. Carlet, E. Prouff, On Plateaued Functions and Their Constructions, *FSE 2003, LNCS 2887,* Springer-Verlag, pp. 57-78, 2003.

[ChPa02] P. Charpin, E. Pasalic, On Propagation Characteristics of Resilient Functions, *SAC 2002, LNCS 2595,* Springer-Verlag, pp. 356-365, 2002.

[CS02] C. Carlet, P. Sarkar, Spectral Domain Analysis of Correlation Immune and Resilient Boolean Functions, *Finite fields and Applications,* Vol. 8, pp. 120-130, 2002.

[CM03] N. Courtois, W. Meier, Algebraic Attacks on Stream Ciphers with Linear Feedback, *Eurocrypt 2003, LNCS 2656,* Springer-Verlag, pp. 345-359, 2003.

[DSS01] Y. Dodis, A. Sahai, A. Smith, On Perfect and Adaptive Security in Exposure Resilient Functions, *Eurocrypt 2001, LNCS 2045,* Springer-Verlag, pp. 301-324, 2001.

[DXS91] C. Ding, G. Xiao, W. Shan, Stability Theory of Stream Ciphers, Springer, 1991.

[J92] D. Jungnickel, *Finite Fields. Structure and Arithmetics,* BI, Wissenschaftverslag, 1992.

[JJ99] T. Johansson, F. Jönsson, Fast Correlation Attacks Based on Turbo Code Techniques, *Crypto 1999, LNCS 1666,* Springer-Verlag, pp. 181-197, 1999.

[K99] K. Kurosawa, Almost Security of Cryptographic Boolean Functions, Cryptology e-print archive, `http://eprint.iacr.org/2003/075`.

[KJS01] K. Kurosawa, T. Johansson, D.R. Stinson, Almost k-wise Independent Sample Spaces and Their Cryptologic Applications, *Journal of Cryptology,* Vol. 14(4), pp. 231-253, 2001.

[KS97] K. Kurosawa, T. Satoh, Design of SAC/PC(ℓ) of order k Boolean Functions and Three Other Cryptographic Criteria, *Eurocrypt 1997, LNCS 1233,* Springer-Verlag, pp. 434-449, 1997.

[KW93] M. Karchmer, A. Wigderson, On Span Programs, *Proc. of 8-th Annual Structure in Complexity Theory Conference,* pp. 102-111, 1993.

[MS] F.J. MacWilliams, N.J. Sloane, *The Theory of Error Correcting Codes,* North Holland, Amsterdam, 1977.

[M69] J.L. Massey, Shift-Register Synthesis and BCH Decoding, *IEEE Trans. Information Theory,* pp. 122-127, 1969.

[MS92] W. Meier, O. Staffelbach, Fast Correlation Attacks on Certain Stream Ciphers, *Journal of Cryptology,* pp. 67–86, 1992.

[NN03] V. Nikov, S. Nikova, On a Relation Between Verifiable Secret Sharing Schemes and a Class of Error-Correcting Schemes, Cryptology e-print archive, `http://eprint.iacr.org/2003/210`.

[PVV+91] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, J. Vandewalle, Propagation Characteristics of Boolean Functions, *Eurocrypt 1990, LNCS 473,* Springer-Verlag, pp. 161-173, 1991.

[RS87] R.A. Rueppel, O.J. Staffelbach, Products of Linear Recurring Sequences with Maximum Complexity, *IEEE Trans. Information Theory,* Vol. 33(1), pp. 124–131, 1987.

[S84] T. Siegenthaler, Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications, *IEEE Trans.Inf.Theory,* 776–780, 1984.

[S85] T. Siegenthaler, Decrypting a Class of Stream Ciphers Using Ciphertext Only, *IEEE Trans. Computers,* pp. 81–85, 1985.

[SM00] P. Sarkar, S. Maitra, New directions in Design of Resilient Boolean Functions, Cryptology e-print archive, `http://eprint.iacr.org/2000/009`.

[XM88] G.Z. Xiao, J.L. Massey, A spectral Characterization of Correlation-Immune Combining Functions, *IEEE Trans.Inf.Theory,* Vol. 34, pp. 569-571, 1988.

[ZZ97] Y. Zheng, X.M. Zhang, Cryptographically Resilient Functions, *IEEE Trans. Information Theory*, Vol. 43(5), pp. 1740-1747, 1997.

[ZZ99b] Y. Zheng, X.M. Zhang, Plateaued Functions, *Int.Conf. on Inf. and Communic. Security, LNCS 1726,* Springer-Verlag, pp. 284-300, 1999.

[ZZI99] Y. Zheng, X.M. Zhang, H. Imai, Connections Between Nonlinearity and Restrictions, Terms and Hypergraphs of Boolean Functions, *IEEE International Symposium on Inform. Theory 1998,* IEEE Press, pp. 439, 1998.

[ZZ00] Y. Zheng, X.M. Zhang, On Relationship Among Avalanche, Nonlinearity, and Propagation Criteria, *Asiacrypt 2000, LNCS 1976,* Springer-Verlag, pp. 470-483, 2000.

Information Theory and the Security of Binary Data Perturbation

Poorvi L. Vora

George Washington University, Washington DC 20052
poorvi@gwu.edu

Abstract. Random data perturbation (RDP) has been in use for several years in statistical databases and public surveys as a means of providing privacy to individuals while collecting information on groups. It has recently gained popularity as a privacy technique in data mining. To our knowledge, attacks on binary RDP have not been completely characterized, its security has not been analyzed from a complexity-theoretic or information-theoretic perspective, and there is no privacy measure of binary RDP that is related to the complexity of an attack. We characterize all inference attacks on binary RDP, and show that if it is possible to reduce estimation error indefinitely, a finite number of queries per bit of entropy is enough to do so. We define this finite number as the privacy measure of the binary RDP.

1 Introduction

The general problem solved by random data perturbation (RDP) is that of providing statistics while protecting individual values. This is done by adding noise to the individual values. The larger the probabilistic perturbation of the data, the more privacy provided to the individual values, and the less accurate the statistics. RDP has been in use for about twenty years in statistical database security [1, 12], and has recently been proposed as a means of personal privacy protection in data mining applications [2, 3]. In this paper, we analyze the amount of privacy provided by binary RDP.

The purpose of a statistical database is to provide statistics to researchers while keeping individual values "private". For example, a health database would keep "private" whether individual X had Hepatitis A or not, but would reveal how many members in a community had Hepatitis A. The general technical problem is as follows:

- Database A contains two (possibly intersecting) sets of binary variables: $\mathcal{Q} = \{q_1, q_2, ...q_i, ...\}$ (queryable bits) and $\mathcal{S} = \{s_1, s_2, ...s_i, ...\}$ (sensitive bits).
- Data collector B queries the value of $f(a_1, a_2, ..., a_k)_{a_i \in \mathcal{Q}} = X \in \{0, 1\}$, for *any* f. In particular, B can query combinations of queryable values across records such as "the most common gender among records 1, 2 and 3".

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 136–147, 2004.

- $\mathcal{Q}$ and $\mathcal{S}$ are probabilistically-related, i.e. the mutual information between the two (the change in uncertainty in one on knowing values in the other) is non-zero: $\mathcal{I}(\mathcal{S};\mathcal{Q}) \neq 0$.
- The bits in $\mathcal{S}$ are to be "protected".

One approach is to compute the value of $f(a_1, a_2, ...a_k)$ so that no other information is revealed. This can be done using secure multiparty computation [9]. However, secure multiparty computation cannot prevent *inference attacks* [6, 12], which involve the determination of information on bits in $\mathcal{S}$ from several queried values $A_i = f_i(a_1, a_2, ...a_k)$, $i = 1, 2, ...n$. It is not straightforward to recognize such attacks.

Example 1. The general inference attack. Consider
s_1 "gender"
s_2 "Over 40"
q_1 "Losing Calcium"
q_2 "Balding"
q_3 "Greying"
q_4 "Gaining weight"

Suppose B wishes to determine bits s_1 and s_2. To do so, B may query functions of the bits q_1, q_2, q_3, q_4, which would reveal information about the sensitive bits, but would not determine them completely. For example, women over 40 are more likely to be losing Calcium than any of the three other categories. Similarly, men over 40 are almost the only category balding. However, it is possible for a man over 40 to have the same responses as a man under 40.

The RDP of $A_i = f_i(a_1, a_2, ...a_k)$ before revealing the values, whether computed using trusted multiparty computation or not, makes the task of inference more difficult. Binary RDP proceeds as follows:

B requests bit X from A and receives the variable $\phi(X) = Y \in \{0, 1\}$ generated according to $P(Y|X)$,

$$P(Y|X) = \begin{cases} \rho & Y = X \\ 1 - \rho & Y \neq X \end{cases}$$

i.e. B receives the requested bit flipped with probability $1 - \rho$. There are no conditions on the amount of information A has about what B received, i.e. we do not consider private information retrieval.

The simplest attack on RDP is the repeated query attack, where B repeatedly asks for the same bit $x \in \mathcal{Q} \cap \mathcal{S}$ and guesses the correct value of x to be the one received most often (assuming $\rho > \frac{1}{2}$). Clearly, the estimation error can be decreased without bound by increasing queries without bound, i.e. if ω_m is the probability of error using m repeated queries of a bit, $\lim_{m\to\infty} \omega_m = 0$.

However, this is the best B can do with repeated queries, i.e. if η_m is the number of queries per bit determined, η_m increases indefinitely if ω_m is not bounded below.

$$\lim_{m\to\infty} \omega_m = 0 \Rightarrow \lim_{m\to\infty} \eta_m = \infty \tag{1}$$

There are a number of attacks other than a repeated query attack, which is very recognizable. However, because these are not well characterized, it is typically assumed that expression (1) represents a best-case scenario for B. In [16] it is stated that, if $\mathcal{C}$ is the channel capacity of the protocol viewed as a communication channel, attacks in which

$$\lim_{m\to\infty} \omega_m = 0 \; for \; \eta_m = \frac{1}{\mathcal{C}} \tag{2}$$

exist, i.e. that inference attacks can be more efficient than the repeated query attack - if the queries $\mathbf{x}$ are functions of the bits B wants to determine, and if B and A are willing to participate in a large enough number of queries.

In this paper we show that expression (2) *is* the best an attacker can do, i.e. that

$$\lim_{m\to\infty} \omega_m = 0 \Rightarrow \lim_{m\to\infty} \eta_m \geq \frac{1}{\mathcal{C}}$$

for all inference attacks if $\lim_{m\to\infty} \eta_m$ exists. Note that the lower bound obviously does not hold for attacks that do not seek to reduce error arbitrarily. We define the asymptotic lower bound on η_m as the privacy measure of the protocol; it is the inverse of the channel capacity of the protocol viewed as a communication channel. Note that, we use "asymptotic" as used by mathematicians, to mean: "in the limit".

When the protocol has a small bias β (i.e. each bit is flipped with probability $0.5 + \beta$, β small), Chernoff-type bounds [11, 10] provide estimates of the query complexity of a repeated query attack. For example, from the Chernoff bound:

$$m = \eta_m = \frac{[ln(\frac{2}{\delta})]}{0.38\beta^2} \Rightarrow \omega_m \; \leq \; \delta$$

We show that η_m for an inference attack can be *independent of* δ, i.e. while $m \to \infty$, η_m can be finite, though bounded below. In particular we show that η_m is $\Theta(\frac{1}{\beta^2})$.

Our main contributions are: (a) the framework we have used to study the security of binary RDP, and the corresponding definitions and associations with information theory and coding; (b) a general characterization of inference attacks, and (c) the use of our framework in deriving a very general efficiency result that changes some of the view of the efficiency of inference attacks.

The paper is organized as follows. In section 2 we present a short review of existing work, and in section 3, definitions motivated by the statistical database security problem. Section 4 presents our results with proofs. The conclusions are presented in Section 5.

2 Related Work

The database community has measures of the privacy of randomization [8, 3, 2]; these are, however, not motivated by a security analysis. The security analyses

that do exist [12] focus on the variance of the estimation error. [2] proposes the use of the differential mutual information between the original and perturbed continuous-valued data points as a measure of "conditional privacy loss", which inspires our measure. The mutual information between two variables is the change in uncertainty of one on knowing the other. Thus the measure of conditional privacy loss has some useful properties: (a) it addresses the *change* due to a protocol instance, and (b) because it is based on entropy, it distinguishes among situations where the two possibilities are almost equally likely and situations where this is not so. The measure does, however, depend on the original pdf, and not only on protocol parameters. Our privacy measure, the inverse of the protocol channel capacity, is closely related to this measure, but improves on it by being independent of the input pdf (channel capacity is the maximum value of the mutual information, taken over all possible input pdfs). Unlike [2], our work also provides a connection between our privacy measure and the complexity of certain types of attacks.

3 Definitions

In this section we provide the definitions we shall need to prove our results. We provide a list of symbols in the appendix.

Consider a query sequence $\mathbf{x}$ of length m. The number of possible values of the true responses need not be 2^m, because certain bit combinations may not be possible, as the queries are not generally independent. We denote the size of the set of all possible values of $\mathbf{x}$ by M. Clearly, a "most efficient" query sequence would use exactly $log_2 M$ bits to distinguish among the M values, but most effective query sequences would want to correct for the RDP and would hence consist of more that $log_2 M$ queries.

Definition 1. *The* query complexity per bit*, of query sequence* $\mathbf{x}$ *of length* m*, as a means of distinguishing among* M *possible values of* $\mathbf{x}$ *is* $\eta_m = \frac{m}{log_2 M}$.

We define the most general inference attack, such as the one of example 1, next.

Definition 2. *An* inference attack *is a set of queries* $\mathbf{x}$ *such that* $\mathbf{x}$ *and the set of sensitive bits* $\mathcal{S}$ *are not independent, i.e.* $I(\mathcal{S}; \mathbf{x}) \neq 0$.

The definition is intentionally broad, as we show a lower bound on the query complexity per bit for an inference attack for which $\lim_{m\to\infty} \omega_m = 0$. The definition also assumes nothing about the relationship between queried bit x_i and previously received responses: $\phi(x_1), \phi(x_2), ...\phi(x_{i-1})$, and, hence includes adaptive inference attacks.

B cannot do any better in reducing the uncertainty of $\mathcal{S}$ than is possible through accurate knowledge of all of $\mathcal{Q}$ and unlimited computing power. Assume, wlog, that B wishes to determine the k bits $\mathbf{p} = (p_1, p_2, ...p_k) = \{g_i(a_1, a_2, ..a_j...)_{a_j \in \mathcal{Q}}\}_{i=1}^{k}$ from each record in database A. The RDP limits B

in determining $\mathbf{p}$ accurately, but does not affect the uncertainty reduction in $\mathcal{S}$ from complete knowledge of $\mathbf{p}$. In evaluating the RDP, hence, we focus on the accurate determination of $\mathbf{p}$. We denote the entropy of $\mathbf{p}$ as *queryable entropy*, (that which can be reduced to zero through queries if there is no RDP). Contrast this to the entropy of bits in $\mathcal{S}$, which, in general, cannot be reduced to zero through queries of functions of queryable bits even if there were no RDP.

The maximum probability of estimation error[1], denoted ω_m, is a measure of the success of query sequence $\mathbf{x}$, of length m, in estimating $\mathbf{p}$. As the value of m grows, it is reasonable to expect the error to reduce, or, at least, not increase. We define attacks in which asymptotic error is zero as *small error* attacks.

Definition 3. *A* small error inference attack *is one in which* $\lim_{m\to\infty} \omega_m = 0$.

Clearly, error and query complexity are related, and a lower error could require a higher query complexity per queryable bit. An RDP that forces a higher query complexity to reduce error is better from the privacy point of view. We propose that the measure of the privacy of binary RDP be the minimum value of the query complexity per bit of queryable entropy required for a small error attack.

Definition 4. *The* privacy *of binary RDP is the (tightest) asymptotic lower bound on the query complexity, on average, per bit of queryable entropy, for a small error attack.*

We now review some definitions from information theory necessary for our results.

Definition 5. *[5] A* communication channel *is a triplet of the following: a set of input variables,* $\mathcal{X}$*, a set of output variables,* $\mathcal{Y}$*, and a* a posteriori pdf*,* $P(Y|X)$*, and is denoted* $(\mathcal{X}, P(Y|X), \mathcal{Y})$.

We denote the channel corresponding to a protocol by Φ, and the channel corresponding to binary RDP with probability of lie $1-\rho$ by $\Phi_{\mathcal{B}}(1-\rho)$.

Definition 6. *The* channel capacity *of protocol* Φ *is the maximum decrease in entropy of variable* X *due to the protocol, and is denoted* $\mathcal{C}(\Phi)$.

The channel capacity of the binary symmetric protocol with probability of a lie $1-\rho$ is

$$\mathcal{C}(\Phi_{\mathcal{B}}(1-\rho)) = 1 - \mathcal{H}(\rho) = 1 + \rho log_2\rho + (1-\rho)log_2(1-\rho)$$

bits, where $\mathcal{H}(\rho)$ is the entropy of the binary variable with ρ being the probability of one of its values. When the protocol has a small bias, i.e. $\rho = 0.5+\beta$ for small β, its capacity is determined by the second order term of the Taylor expansion (zeroth and first order terms are zero):

$$\mathcal{C}(\Phi_{\mathcal{B}}(0.5 \pm \beta)) = \frac{2\beta^2}{ln2}, \beta \text{ small} \tag{3}$$

[1] The estimation error calculation assumes a maximum likelihood estimation.

4 Our Results

We wish to determine the privacy of binary RDP. To do so, we demonstrate an asymptotic lower bound on the query complexity, per bit of queryable entropy, for a zero error inference attack. [16] implies that the bound is tight. However, attacks that achieve the bound might be recognizable.

We approach the problem by viewing binary RDP as a communication channel as in [16]. The analogy with communication over a channel is as follows: the protocol is a channel and $\mathbf{p}$ a message. The channel coding ("Shannon's second") theorem [14, 5] provides a tight upper bound of channel capacity on the inverse of η_m for a zero error attack - if each query is a function of $\mathbf{p}$, and η_m is constant as m increases. Hence, when the query sequence $\mathbf{x}$ is a function of $\mathbf{p}$, inference attacks are channel codes; η_m^{-1} are the rates of the codes; when such attacks are zero-error with constant $\eta_m = \eta$, the inverse of channel capacity is the minimum value of η, achieved by attacks that correspond to Shannon codes.

The most general inference attack (see example 1, section 1 and definition 2, section 3) is not one in which the query sequence is a function of the required values $\mathbf{p}$. Nor does an inference attack require constant η_m as m increases. By modifying the proof of the converse of the channel coding theorem using Fano's inequality [5–pg. 205] - the main ingredient for demonstrating channel capacity as a bound on the rate of a code - we show that the tight asymptotic lower bound on the query complexity per bit for the (more general) small error inference attack is also the inverse of the channel capacity of the protocol. Fano's inequality provides the *asymptotic lower bound* on η_m, and the result in [16] and the channel coding theorem provide the *existence* of zero error inference attacks that achieve it.

Theorem 1. *Given a binary RDP Φ, an asymptotic lower bound on η_m, for a small error inference attack, is $\frac{1}{\mathcal{C}(\Phi)}$. More formally,*

$$\lim_{m\to\infty} \omega_m = 0 \Rightarrow \exists\ \{\Lambda_m\}_{m=1}^{\infty} \text{ such that } \eta_i \geq \Lambda_m \forall i \geq m \text{ and } \lim_{m\to\infty} \Lambda_m = \frac{1}{\mathcal{C}(\Phi)}$$

Proof. The proof is similar to the proof of the converse of the channel coding theorem [5], except for two differences: (a) in an inference attack, queries $\mathbf{x}$ are not necessarily a function of bits required $\mathbf{p}$, and (b) inference attacks do not have constant η_m as m increases.

Assume $\lim_{m\to\infty} \omega_m = 0$, i.e. the attack is small error. Then $\lim_{m\to\infty} E_m = 0$ where E_m is the average probability of error. Consider the case when the values of $\mathbf{p}_m$ are equally likely. Then,

$$log_2 M = \mathcal{H}(\mathbf{p}_m) = \mathcal{H}(\mathbf{p}_m|\phi(x_1), \phi(x_2), ...\phi(x_m)) + I(\mathbf{p}_m; \phi(x_1), \phi(x_2), ...\phi(x_m))$$

From equation (8.95) (Fano's inequality), [5–pg. 205],

$$\mathcal{H}(\mathbf{p}_m|\phi(x_1), \phi(x_2), ...\phi(x_m)) \leq 1 + E_m log_2 M$$

and hence,

$$log_2 M \leq 1 + E_m log_2 M + I(\mathbf{p}_m; \phi(x_1), \phi(x_2), ...\phi(x_m)) \tag{4}$$

Further,

$$
\begin{aligned}
&I(\mathbf{p}_m; \phi(x_1), \phi(x_2), ...\phi(x_m)) \\
&= \mathcal{H}(\phi(x_1), \phi(x_2), ...\phi(x_m)) - \mathcal{H}(\phi(x_1), \phi(x_2), ...\phi(x_m)|\mathbf{p}_m) \\
&= \mathcal{H}(\phi(x_1), \phi(x_2), ...\phi(x_m)) - \textstyle\sum_i \mathcal{H}(\phi(x_i)|\phi(x_1), \phi(x_2), ...\phi(x_{i-1}), \mathbf{p}_m) \\
&\leq \mathcal{H}(\phi(x_1), \phi(x_2), ...\phi(x_n)) - \textstyle\sum_i \mathcal{H}(\phi(x_i)|\phi(x_1), \phi(x_2), ...\phi(x_{i-1}), \mathbf{p}_m, x_i) \\
&= \mathcal{H}(\phi(x_1), \phi(x_2), ...\phi(x_m)) - \textstyle\sum_i \mathcal{H}(\phi(x_i)|x_i) \\
&\leq \textstyle\sum_i \mathcal{H}(\phi(x_i)) - \textstyle\sum_i \mathcal{H}(\phi(x_i)|x_i) \\
&= \textstyle\sum_i I(x_i; \phi(x_i)) \\
&\leq m\mathcal{C}(\Phi)
\end{aligned}
$$

From equation (4),

$$log_2 M \leq 1 + E_m log_2 M + m\mathcal{C}(\Phi)$$

Hence,

$$\eta_m = \frac{m}{log_2 M} \geq \frac{1 - E_m}{\frac{1}{m} + \mathcal{C}(\Phi)} = \Lambda_m$$

and

$$\lim_{m \to \infty} \Lambda_m = \frac{1}{\mathcal{C}(\Phi)}$$

$$\eta_m = \Omega(1)$$

Theorem 2. *For a binary RDP Φ, $\forall \Lambda > \frac{1}{\mathcal{C}(\Phi)}$, there exists a small error inference attack on Φ with $\eta_m = \Lambda$, $\forall m$.*

Proof. Follows from the channel coding theorem [14].

Theorem 1 indicates that $\eta_m = \Omega(1)$. Theorem 2, that $\eta_m = \Theta(1)$.

Attacks that correspond to codes are those where the queries $\mathbf{x}$ are deterministic functions of the desired bits $\mathbf{p}$. These are rare but not impossible. We provide an example of such an attack.

Example 2. The deterministically-related query attack. Consider a database of records of all residents of a county. From each record, consider the set of the following bits:

x_1. "location = North";
x_2. "virus X test = positive";
x_3. "gender = male" AND "condition Y = present".

Suppose it is also known that, for this county,

$$(location \ = \ North) \oplus (virus \ X \ test \ = \ positive) \Leftrightarrow (gender \ = \ male) AND \ (condition \ Y \ = \ present) \tag{5}$$

i.e,

$$x_1 \oplus x_2 \ = \ x_3 \tag{6}$$

for all records, where $\oplus$ represents the XOR operation. This could be determined, for example, from county health statistics.

Suppose B chooses as desired bits $\mathbf{p} = (x_1, x_2)$ for all records, and designs an over-determined query sequence by also requesting x_3. Without randomization, B would not need to do so; with randomization, x_3 serves as a parity check for the values of x_1 and x_2, or, in the communication channel framework, as an error-detecting symbol. The queries $\mathbf{x} = (x_1, x_2, x_3)$ may be thought of as the code bits. In general, one can have an over-determined sequence of m queries whose values are completely determined by $\mathbf{p}$ - through a set of m equations known to be satisfied by $\mathbf{p}$ and $\mathbf{x}$. Equation (6) is one such equation.

If the attack is recognized, A could:

(a) refuse to respond
(b) respond with $\phi(x_1) \oplus \phi(x_2)$ instead of $\phi(x_1 \oplus x_2)$.

Recognizing the attack is not trivial. If, instead of "male with condition Y", x_3 were, "$(location = North) \oplus (virusXtest = positive)$", it may be recognized by A, through extensive record keeping, as a logical combination of previously provided bits. But in the form of a request for a bit about gender and condition Y, and in the absence of knowledge of the specific relationship of equation (5), or a causal relationship - as opposed to a statistical one in a limited population - gender and condition Y are not readily seen to be revealing information regarding infection with virus X. Such an attack is fairly difficult to recognize, and hence to counter.

An approach like that of the source-channel coding theorem shows that B cannot do better using another procedure. This gives our final result, that the tight asymptotic lower bound on query complexity for zero asymptotic error is the ratio of queryable entropy to protocol channel capacity. As a corollary, the privacy of binary RDP is the inverse of its channel capacity.

The values of $\mathbf{p}_m$ are not necessarily uniformly distributed, and hence the entropy of $\mathbf{p}$, the queryable entropy, is not necessarily $log_2 M$. From the source coding theorem, if the entropy of $\mathbf{p}$ is $\mathcal{H}(\mathbf{p})$, then $\mathbf{p}$ is represented by $\mathcal{H}(\mathbf{p})$ bits on average (over many records). This observation can be combined with a reasoning similar to that in Theorem 1 to obtain a result similar to that of the source-channel coding theorem, except, as with Theorem 1, inference attacks are not of constant η_m, and do not consist of queries $\mathbf{x}$ that are deterministic combinations of the required bits $\mathbf{p}$. Again, we derive the asymptotic lower bound, and Shannon's results show it is tight.

Theorem 3. *The tight asymptotic lower bound on the query complexity, on average, per record, for a small error inference attack, is* $\frac{\mathcal{H}(\mathbf{p})}{\mathcal{C}(\Phi)}$ *if the record sequence is stationary, i.e. if the number of records is* N_r*, and* γ_m *the number of queries per record of sequence* $\mathbf{x}$*,*

$$\lim_{m\to\infty} \omega_m \to 0 \Rightarrow \exists\ \Gamma_m \text{ such that } \gamma_m \geq \Gamma_m\ \forall\ i \geq m \text{ and } \lim_{N_r\to\infty} \Gamma_m = \frac{\mathcal{H}(\mathbf{p})}{\mathcal{C}(\Phi)}$$

Proof. $\frac{\mathcal{H}(\mathbf{p})}{\mathcal{C}(\Phi)}$ *is an asymptotic lower bound*: Assume the existence of a small error attack with asymptotic query sequence length $K = \frac{\mathcal{H}(\mathbf{p})}{\mathcal{C}(\Phi)} - \Delta$ per record on average, $\Delta > 0$. This means that, given $\epsilon, \delta > 0$, a query sequence of length at

most $m = N_r(K+\epsilon)$ for N_r records, N_r large enough, can result in a probability of error at most δ. By Theorem 1, for any given ν, η_m for the attack must be at least $\frac{1}{\mathcal{C}(\Phi)} - \nu$, for large enough m, and hence the length of $\mathbf{p}$, $\frac{m}{\eta_m}$, at most $\frac{N_r(K+\epsilon)}{(\frac{1}{\mathcal{C}(\Phi)}-\nu)} = \frac{N_r(\mathcal{H}(\mathbf{p})-\mathcal{C}(\Phi)(\Delta-\epsilon))}{1-\nu\mathcal{C}}$, i.e. each record is represented, on average, by a number of bits strictly smaller than the record entropy for small enough ϵ, δ, ν. This violates Shannon's source coding theorem [5–pg. 89, Thm. 5.4.2] and [14].

$\frac{\mathcal{H}(\mathbf{p})}{\mathcal{C}(\Phi)}$ *can be achieved from above (i.e. tightness)*: straightforward from Shannon's source-channel coding theorem [5].

Thus Theorem 3 says that the query complexity per record, on average, for a zero error attack, is independent of the error.

Theorem 2 says that small error attacks in which η_m remains the same (but decrease in error is paid for by increase in total query complexity) exist if $\eta_m \geq \frac{1}{\mathcal{C}(\Phi)}$. It does not say anything about how the attacks will be constructed, and while the query complexity is tightly bounded below, the information-theoretic result does not indicate whether the processes of determining the values of $\mathbf{x}$ and $\mathbf{p}$ are computationally feasible. Recall that the value of $\mathbf{x}$ is computed by the database, A, and its complexity is measured by the number of logical operations performed to produce a response to a query from points in the database.

Some results since Shannon's work help address the issue of feasibility and construction. Forney's work, originally published in [7], shows that Shannon codes that are encodable and decodable in polynomial time exist. This implies that polynomial-time small-error attacks of constant finite η_m exist. More recent work, that of Spielman, [15] shows how to construct linear time encodable and decodable codes that approach the channel coding theorem's limits. Thus, linear time attacks with η_m approaching η_{min}, and arbitrarily low error, can be constructed. It is likely that attacks modeled on good, computationally feasible, error-correcting codes would consist of queries $\mathbf{x}$ that are rather contrived combinations of queryable bits from $\mathcal{Q}$. It is not clear how easy it would be to recognize such attacks. Recognizability constraints, ignored by us, could affect the existence result.

Corollary 1. *The tight asymptotic lower bound on the query complexity, per bit of queryable entropy, for a small error inference attack on* $\Phi_{\mathcal{B}}(0.5 \pm \beta)$ *is* $\frac{ln2}{2\beta^2}$. *Hence* η_m *is* $\Theta(\frac{1}{\beta^2})$.

Proof. The result follows from Theorem 1-3 and equation (1).

Corollary 2. *The privacy of* Φ *is* $\frac{1}{\mathcal{C}(\Phi)}$.

Proof. Follows from Theorems 1-3 and Definition 4.

Corollary 3. *The privacy of* $\Phi_{\mathcal{B}}(0.5 \pm \beta)$ *is* $\Theta(\frac{1}{\beta^2})$.

Proof. Follows from Corollary 1 and Definition 4.

In statistical databases, it is typically assumed that a larger number of queries (per attribute desired) is required for a lower error. Our proof of the existence of small error attacks for all asymptotic rates below channel capacity implies that a finite, fixed number of queries, per attribute desired, can ensure asymptotic error is zero; i.e. while total cost needs to increase to reduce error, the cost per bit of entropy need not.

Further, our work demonstrates that some inference attacks, which may not be as recognizable as repetition queries, are less expensive per bit. Last, at first glance it might appear that combinations of a greater number of bits for a query provides greater protection of the bits. But we have shown that combinations of a greater number of bits may also reduce error considerably through B's use of efficient error correcting codes.

Though our results follow very easily from classical results in information theory and coding, our view of the protocol as a channel has one important point of difference from the view of a channel in communication theory. The goal of communication theory is to increase information transfer over a channel given certain constraints. The goal of a privacy protocol is to decrease the information transfer over the protocol given certain constraints (such as the error in statistics that use these perturbed data points). Because of this, A would be interested in channels with small capacity, i.e. "good" privacy protocols. On the other hand, B is interested in the efficient transfer of bits over a particular protocol, typically a channel with small capacity, and a number of the constructive results from the theory of coding are of interest to him.

5 Conclusions

Our result on the correspondence between channel codes and certain types of inference attacks is an example of the study of attacks on non-perfect protocols using results from coding theory. Interesting further results could follow from viewing non-perfect anonymous delivery protocols - such as Crowds [13] and non-perfect combinations of mixes - as channels. Ramp secret sharing schemes [4] might also be amenable to this approach. An even more interesting direction of further work is to determine if our approach provides ingredients for a theory of statistical attacks on block and stream ciphers known to leak information.

References

1. Nabil R. Adam and John C. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Computing Surveys*, Vol. 21, No. 4, pp. 515-556, December 1989.
2. D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. *Proceedings of the Twenteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Santa Barbara, California, USA, May 21-23 2001.

3. R. Agrawal and R. Srikant. Privacy-Preserving Data Mining. *Proc. of the ACM SIGMOD Conference on Management of Data*, Dallas, May 2000.
4. G. R. Blakley and C. Meadows. Security of ramp schemes. *Proc. of Crypto'84*, Lecture Notes on Comput. Sci., 196, pp. 242–268, Springer Verlag, 1984.
5. Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley and Sons, 1991.
6. Csilla Farkas and Sushil Jajodia. The inference problem: a survey. *ACM SIGKDD Explorations Newsletter* Volume 4, Issue 2, pp. 6-11, December 2003.
7. David G. Forney. *Concatenated Codes*, MIT Press, Cambridge, Mass., 1966.
8. Diane Lambert. Measures of Disclosure Risk and Harm. *Journal of Official Statistics*, 9, pp. 313-331, 1993.
9. Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. *Journal of Cryptology*, 15 (3), 177-206, 2002.
10. Michael Luby. *Pseudorandomness and cryptographic applications*. Princeton Computer Science Notes, 1996.
11. Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*, pp. 67-73, Cambridge University Press, New York, NY, 1995.
12. Krishnamurty Muralidhar and Rathindra Sarathy. Security of random data perturbation methods. *ACM Transactions on Database Systems (TODS)* vol. 24, no. 4, Dec. 1999, pp. 487 - 493
13. Michael K. Reiter and Aviel Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, Vol. 1, No. 1, pp. 66-92, November 1998.
14. Claude Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, vol. 27, pp. 379-423, July 1948.
15. Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, Vol 42, No 6, pp. 1723-1732, 1996.
16. Poorvi Vora. The channel coding theorem and the security of binary randomization. Proc., 2003 IEEE International Symposium of Information Theory, Yokohama, Japan, June 30 - July 4, pp. 306, 2003.

A Appendix: List of Symbols

A	Database
B	Data collector
$\mathcal{Q}$	set of queryable bits
q_i	a queryable bit
$\mathcal{S}$	set of sensitive bits
s_i	a sensitive bit
X, x_i, A_i	a single queried bit
$\mathcal{I}(\alpha;\beta)$	the mutual information between α and β
$Y = \phi(X)$	a single response to a query X
ρ	probability of truth
Σ	$\{0,1\}$
$P(Y\|X)$	posterior pdf, (or *a posteriori* pdf) of protocol/channel
m	number of queries or length of query sequence $\mathbf{x}$
ω_m	maximum probability of error for $\mathbf{x}$
η_m	ratio of queries to bits determined for $\mathbf{x}$

$\mathbf{x}$	a query sequence
M	number of values of $\mathbf{x}$
$\mathbf{p}$	sequence of queryable bits B wishes to determine
k	number of required bits or length of $\mathbf{p}$
Φ	protocol/channel
$\mathcal{C}(\Phi)$	capacity of Φ
$\Phi_{\mathcal{B}}$	binary protocol
β	small bias of a binary protocol
E_m	average probability of error of protocol using $\mathbf{x}$
$\mathcal{H}$	entropy of $\mathbf{q}$
N_r	number of records

Symmetric Authentication Codes with Secrecy and Unconditionally Secure Authenticated Encryption

Luke McAven[1], Reihaneh Safavi–Naini[1], and Moti Yung[2]

[1] School of Information Technology and Computer Science, University of Wollongong, Wollongong, NSW 2522, Australia
{lukemc, rei}@uow.edu.au
[2] Department of Computer Science, Columbia University, New York, NY 10027, USA
moti@cs.columbia.edu

Abstract. Unconditional security provides security independent of assumptions regarding adversaries resources. Considerable research has been carried out into unconditionally secure authentication codes without secrecy, wherein the confidentiality of the plaintext is unimportant. Unconditionally secure encryption has been less thoroughly studied. The traditional framework for considering integrity and confidentiality in an unconditionally secure environment is that of authentication codes with secrecy. We extend this framework, in the symmetric case, to encompass aspects of recent work on unconditionally secure formulations of authentication codes and encryption systems. This will allow for a systematic analysis of unconditionally secure authenticated encryption schemes.

1 Introduction

There exist scenarios where communicating parties cannot reliably know the resources of potential adversaries. Typically this is true for small mutually distrusting groups of entities that do not know each others computational or technological advantages, e.g. advances in quantum computations, as is the setting between nations. In such cases unconditionally secure (US) schemes provide security independent of resources of the adversary. Furthermore, one can obtain information theoretic bounds on the probability of adversaries succeeding in particular attacks, and combinatorial bounds on the relative sizes of system parameters.

The two main goals of security are authenticity and confidentiality. Unconditional security has its roots in Shannon's theory of unconditional secrecy [19], with the goal of confidentiality. Shannon's original framework and later extensions [25] considered ciphertext only attacks where the adversary observes ciphertexts and tries to learn the associated plaintext. US encryption was extended [12] to include *adaptive adversaries* who access encryption and decryption oracles, bringing US models closer to computationally secure ones.

US authentication codes (A–codes) were first considered by Gilbert *et al.* [7] and later formalised [22] and extended [10, 11, 15, 24]. The main attack considered in authentication systems is *spoofing of order* L where the adversary has

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 148–161, 2004.

access to L authenticated messages and wants to construct a fraudulent message acceptable to the verifier. US authentication was extended [16] to include adaptive attackers with access to authentication and/or verification oracles. [16] developed a general framework for "general group–based unconditionally secure authentication codes" that extends traditional A–codes so as to encompass the strong notions of security necessary for US digital signatures [8, 20].

Simmons [22] model of US A–codes provides both secrecy and authentication. Bounds and characterisations of such *A–Codes with Secrecy* have been given [3, 4, 25, 27], as well as numerous constructions [26]. We consider A–codes with secrecy under strong attacks where the attacker accesses oracles in the manner of [12, 16]. We evaluate security of such codes for secrecy and authenticity goals, and, following computational security, we refer to the systems as *authenticated encryption (AE) systems.*

We restrict ourselves to *symmetric* US authenticated encryption, with the aim of systematically analysing A–codes with secrecy against strong attacks. The approach is, however, extendible to asymmetric and group based systems. We follow [12] in modelling perfect secrecy, and extend this to the integrity related attacks. We give some relationships between security notions, and relate some AE notions to those in classical A–codes.

Several questions motivate us. We would like to find the relationship between AE systems under strong attacks and A–codes and US secrecy codes, both under strong attacks. We would also like to examine security of the composite constructions; MAC–then–encrypt, encrypt–then–MAC and MAC–and–encrypt which can all be seen as A–codes with secrecy. We would also like to construct AE systems with the 'best' performance.

Related Work

Shikata *et al* [21] considered models for US authentication encryption, and analysed methods for composing US secure encryption and US authentication schemes. They built on [9], who considered a US analogy of public key encryption, which they called Unconditionally Secure Asymmetric Encryption (USAE). In parallel with developments in public key cryptography, they considered goals associated with secrecy and *non–malleability* (NM). These papers all consider the scenario that multiple receivers and that adversaries may corrupt some of them. We focus on two party systems and develop the relationship between different notions of security for AE systems (A–codes with secrecy).

The rest of this paper is structured as follows. In section 2 we introduce authentication codes with secrecy, and consider the oracles used in modelling authentication and encryption. Section 3 considers security notions for AE. Section 4 contains some bounds, old and new, relevant to AE systems. In section 5 we describe three methods of composing AE systems from an A–code and an encryption scheme. Section 6 contains a directly constructed AE scheme. We summarise our results in section 7.

2 US Authentication Codes with Secrecy

An A–code with secrecy satisfies the two goals of secrecy and authenticity, hence can be seen as an AE system. The following definition reflects.

An *authentication code with secrecy* [26](or an *authenticated encryption system*) is a tuple $\mathtt{A_S} = (\mathcal{S}, \mathcal{M}, \mathcal{K}, \mathcal{E}, \mathcal{D})$. Each tuple element is a set: $\mathcal{S}$ of plaintexts (source states), $\mathcal{M}$ of ciphertexts (messages sent over the channel), $\mathcal{K}$ of keys, $\mathcal{E}$ of *authenticated encryption functions*, each an injective mapping from $\mathcal{S}$ to $\mathcal{M}$, while $\mathcal{D}$ is the set of *verified decryption functions*. The sets $\mathcal{E}$ and $\mathcal{D}$ are indexed by elements of $\mathcal{K}$, so E_k is the AE function in $\mathcal{E}$ associated with $k \in \mathcal{K}$, and D_k is the verified decryption function in $\mathcal{D}$ associated with $k \in \mathcal{K}$. We have $E : \mathcal{K} \times \mathcal{S} \Rightarrow \mathcal{M}$, $E_k(s) = m$, and $D : \mathcal{K} \times \mathcal{M} \Rightarrow \mathcal{S}$, $D_k(m) = s$ if $E_k(s) = m$. That is the encryption and decryption functions have the property that, $\forall s \in \mathcal{S}$, $m \in \mathcal{M}$, we have $D_k(E_k(s)) = s$. This implies $|\mathcal{M}| \geq |\mathcal{S}|$.

We abbreviate the processes of authenticated encryption and verified decryption to `AuthEnc` and `VerDec`, respectively.

We assume an *a priori* probability distribution $P_{\mathcal{S}}$ on plaintexts, and a probability distribution on the key space, both of which are public. We let $\mathcal{M}(k)$ be a subset of $\mathcal{M}$ generated by $E_k(s), \forall s \in S$. A plaintext, ciphertext pair (s, m) is *valid* if there exists a key k for which $E_k(s) = m$.

The $\mathcal{E}$ functions may be probabilistic. That is, given $k \in \mathcal{K}$ and $s \in \mathcal{S}$, the ciphertext could be one of a set of possible values. Such probabilistic A–codes are called *A–codes with splitting* [5]. The splitting strategy of the sender determines the ciphertext used for a given k and s. Functions in $\mathcal{D}$ are deterministic however, so for a given k and m, always output s.

Authenticated Encryption in Computational Security

AE schemes are used in computationally secure settings [1]. Generic AE gives the outcome of a verified decryption function as a valid source state, or a statement $\perp$ that verification has failed. The process by which this result comes about is treated as a black box. If an AE scheme is constructed as a composite of an authentication and an encryption scheme, however, this outcome is usually achieved through one of two methods. In the first, one decrypts the $m \in \mathcal{M}$ and then verifies it, while in the second, one verifies the $m \in \mathcal{M}$ and then decrypts it if verification succeeds.

While computationally security adopts an inviolable black box, we may consider side–channels allowing testing of component algorithms independently. We discuss this further in section 3.4.

2.1 Modelling Security for A–Codes and US Encryption

Attacks in A–codes with secrecy [22] are traditionally [26] taken as spoofing attacks, where an adversary passively observes the channel then attempts to construct a fraudulent message. In spoofing of order L the adversary observes L ciphertexts. Impersonation and substitution [22] are spoofing of order 0 and 1,

respectively. The secrecy property of A–codes have also been considered [10, 22, 23, 25].

The traditional goal of an adversary in an encryption scheme was to obtain information about the plaintext of a given ciphertext. In assessing these systems one assumed the adversary has access to L ciphertexts.

Security evaluation of A–codes and US encryption systems under strong attacks was considered in [12, 16]. In these attacks the adversary has access to *oracles* implementing system algorithms with participant key information. An oracle receives a query and produces a response. For each query the adversary may use the result of all previous queries. In other words the adversary *adaptively* constructs queries. It has been observed [16] that adaptivity in the construction of queries does not increase the power of the attack in the US setting.

[12, 21] introduced non–malleability as a goal for US encryption systems. In encryption systems *secrecy* relates to observers being unable to gain information about the plaintext associated with an observed ciphertext and is a passive attack. *Non–malleability* relates to observers being unable to modify a challenge ciphertext so as to produce a new ciphertext where the plaintexts of the ciphertexts are related in a known way.

Oracles and oracle queries: [16] developed a framework for security evaluation of generalised A–codes. They used oracles to model adversaries power. For such codes two oracles types were defined.

Authentication oracles (A–oracles) implement the authentication algorithm with the signer's key. When presented with an *Authentication query (A–query)*, consisting of a source state $s \in \mathcal{S}$, the A–oracle generates the authenticated message $m \in \mathcal{M}$.

Impersonation and substitution attacks in traditional A–codes correspond to the case that 0 and 1 A–queries are allowed, respectively.

Verification oracles (V–oracles) implement the verification algorithm with the verifying key. On input $m \in \mathcal{M}$, the V–oracle generates a `True/False` result. The queries to this oracle are called *V–queries*.

In [12], US encryption systems were considered under strong attacks where the adversary may access encryption and/or decryption oracles.

Encryption (Decryption) oracles ($E(D)$–oracles) implement the encryption (decryption) algorithm with the senders (receivers) key. When presented with an *encryption (decryption) query ($E(D)$–query)*, consisting of a plaintext (ciphertext) $s \in \mathcal{S}$ ($m \in \mathcal{M}$), the $E(D)$–oracle generates the corresponding ciphertext (plaintext).

In this definition the message space is without redundancy and any message is a possible output of the source. With redundancy a D–oracle returns $\perp$ if no s corresponding to the query m exists.

3 Security Notions for Authenticated Encryption

We consider notions of security obtained by pairing a security goal and an attack [13]. A **goal** represents what the attacker is trying to achieve, while an **attack** represents the type of information available to the attacker.

In AE schemes we need to consider notions of security associated with both confidentiality and integrity. We use A∧B to indicate a system satisfies security notions A and B.

Goals related to confidentiality: [12] defined notions of security for US encryption with two goals of secrecy and non–malleability. In section 3.1 we recall and adapt parts of that formalism relating to secrecy.

Goals related to integrity: [2] define two integrity related goals:

Ciphertext integrity (`IntC`) provides protection against an adversary who attempts to produce an *acceptable ciphertext* which isn't the result of a query or observation.
Plaintext integrity (`IntP`) provides protection against an adversary who attempts to produce an *acceptable ciphertext* where neither the ciphertext nor the decryption of it are the result of a query or observation.

Oracles: For AE we follow [2] in adopting oracles which implement the algorithms `AuthEnc` and `VerDec`.

`AuthEnc` **oracles (***AE***–oracles)** implement `AuthEnc` with the sender's key. When presented with an `AuthEnc`*query* (*AE*–query), consisting of a plaintext $s \in \mathcal{S}$, the *AE*–oracle generates the ciphertext m.

`VerDec` **oracles (***VeD***–oracles)** implement `VerDec` with the receiver's key (which is the same as the sender's in a symmetric system). When presented with an `VerDec` *query* (*VeD*–query), consisting of a ciphertext $m \in \mathcal{M}$, the *VeD*–oracle generates a plaintext $s \in \mathcal{S}$ or $\perp$ if there is no valid (s, m) pair under receiver's key.

We also let VeD^* be an oracle which returns $\perp$ if the VD oracle returns $\perp$, and $\top$ otherwise. In section 3.4 we consider other oracles.

Attack models: We consider three attack models, adapted to unconditional security from the symmetric computationally secure setting [2].

Ciphertext only (COA$_L$): This corresponds to the adversary observing the output of an *AE*–oracle, but not the input. The net result is a set of L ciphertexts. As Stinson [25] did, for US encryption, we consider only the set, and not the order of ciphertexts. For $L = 1$ this directly extends Shannon's perfect secrecy model.

Chosen plaintext (CPA$_L$): The adversary has access to an *AE* oracle, submits L plaintexts, and receives L associated ciphertexts.

Chosen ciphertext ($\mathbf{CCA}_L$): The adversary accesses a VeD–oracle, submits L ciphertexts, and receives the corresponding plaintexts.

One may also consider the adversaries to have access to some combination of these queries. For example, one could consider a (COA_4,CPA_2) attack where the adversary observes four ciphertexts and has two queries to an AE–oracle. One defines security against a fixed set of query types.

3.1 Secrecy in US Encryption

We follow [12] in defining the *source view* and *advantage* of an adversary.

We define the *knowledge set* $\mathcal{R}$ of an adversary to be the set of oracles queries and responses, and also the set of observations of the output of AE–oracle (that is without having access to the corresponding plaintext).

Definition 1. *We say the* **source view** $P_{\mathtt{A_S},\mathcal{S}}(\mathcal{R})$ *of an adversary, given the knowledge set* $\mathcal{R}$*, is the conditional probability distribution on the plaintexts in* $\mathcal{S}\setminus\mathcal{R}$*. Here* $\mathtt{A_S}$ *denotes authentication with secrecy. The source view is a probability distribution* $P_{\mathtt{A_S},\mathcal{S}}(\mathcal{R}) : \mathcal{S}\setminus\mathcal{R} \to [0,1]$ *satisfying*

$$\sum_{\tilde{s}\in\mathcal{S}\setminus\mathcal{R}} P_{\mathtt{A_S},\mathcal{S}}(\tilde{s}|\mathcal{R}) = 1 \ .$$

Definition 2. *The* **advantage** $\mathbf{Adv}^{\mathtt{PS-ATT}}_{\mathtt{A_S}}(\mathcal{R})$ *of an adversary with* $\mathcal{R}$ *is defined as, for challenge ciphertext* m*,*

$$\mathbf{Adv}^{\mathtt{PS-ATT}}_{\mathtt{A_S}}(\mathcal{R}) = \max_{\tilde{s}\in\mathcal{S}} \left| \frac{P_{\mathtt{A_S},\mathcal{S}\setminus\mathcal{R}}(\tilde{s}|m,\mathcal{R})}{P_{\mathcal{S}\setminus\mathcal{R}}(\tilde{s})} - 1 \right| \ .$$

In $\mathbf{Adv}^{\mathtt{PS-ATT}}_{\mathtt{A_S}}(\mathcal{R})$ we replace ATT by the attack, specifying one of the attacks defined earlier. $\mathtt{A_S}$ denotes A–codes with secrecy.

We define ϵ–perfect and perfect ($\epsilon = 0$) secrecy following [12].

Definition 3. *An US AE system provides* ϵ–**perfect secrecy** *under an attack* ATT *if* $\mathbf{Adv}^{\mathtt{PS-ATT}}_{\mathtt{A_S}} = \max_{m\in\mathcal{M}} \mathbf{Adv}^{\mathtt{PS-ATT}}_{\mathtt{A_S}}(m;\mathcal{R}) \le \epsilon$.

3.2 Modelling `IntC` and `IntP`

The computational advantage of an adversary for `IntC` is calculated [2] by determining the probability that a randomly chosen message is acceptable under the key k. We model the US analysis of `IntC` using the view and advantage notions. We firstly define the *Validity View*, similar to the source view but associated with the ciphertext space. We use $V_k(m)$ to represent the result of submitting $m \in \mathcal{M}$ to the VeD–oracle with key k.

Definition 4. *The* **Validity view** $P_{\mathtt{A_S},\mathcal{M}}(\mathcal{R})$ *of an adversary, given a knowledge set* $\mathcal{R}$*, is the distribution on the message space in* $\mathcal{M}\setminus\mathcal{R}$*. The validity view is a probability distribution* $P_{\mathtt{A_S},\mathcal{M}}(\mathcal{R})$ *where* $P_{\mathtt{A_S},\mathcal{M}}(m)$*,* $m \in \mathcal{M}$*, is the probability of* m *being valid* $\mathcal{R}$*.*

$$P_{\mathtt{A_S},\mathcal{M}}(m \textit{ is valid } |\mathcal{R}) = \sum_{k\in\mathcal{K}:VeD_k^*(m)=\top} P(k|\mathcal{R}) \ .$$

Since decryption must be unambiguous, every plaintext goes to a distinct message. For any key there exist $|\mathcal{S}|$ elements of $|\mathcal{M}|$ which are valid under that key. To define the advantage of an adversary in breaking the integrity of a system, following [2], we consider two types of attackers denoted A_{ctxt} and A_{ptxt}. Both attackers have similar access to AE and VeD oracles, however their success is defined differently.

Attackers may query AE–oracle L_1 times (CPA_{L_1}), VeD–oracle L_2 times (CCA_{L_2}) and observe the output of AE–oracle L_3 times.

Let $\mathcal{R}_{AE} = \{(s_1, m_1)\cdots(s_{L_1}, m_{L_1})\}$ denote the knowledge set resulting from querying AE–oracle L_1 times. Similarly, from the L_2 queries to the VeD–oracle we obtain $\mathcal{R}_{VeD} = \{(m_1^v, r_1), \cdots (m_{L_2}^v, r_{L_2}\}$, $r_i \in \mathcal{S}\bigcup\{\perp\}$. Finally we have $\mathcal{R}_{AE}^o = \{m_1^o)\cdots m_{L_3}^o\}$ consisting of observing L_3 outputs of AE–oracle without having access to the corresponding input. Then $\mathcal{R} = \mathcal{R}_{AE}\bigcup\mathcal{R}_{VeD}\bigcup\mathcal{R}_{AE}^o$.

Attacker A_{ctxt} succeeds if they construct a ciphertext $\tilde{m}$ such that (i) The VeD^*–oracle outputs $\top$, and (ii) $\tilde{m} \notin \mathcal{R}$. That is $\tilde{m}$ is a valid ciphertext not in the knowledge set.

Attacker A_{ptxt} succeeds if they construct a ciphertext $\tilde{m}$ such that (i) VeD^*–oracle outputs $\top$, (ii) $\tilde{m} \notin \mathcal{R}$, and (iii) $\tilde{s} = \mathtt{VerDec}(\tilde{m}) \notin \mathcal{R}$. Here the attacker needs to generate a valid ciphertext such that neither the ciphertext nor it's decryption are in the knowledge set.

We define the *advantages* of the above adversaries as

$$\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntC-ATT}}(\mathcal{R}) = \max_{\tilde{m},m'} P_{\mathtt{A_{ctxt}},\mathcal{M}\setminus\mathcal{R},}(\tilde{m} \text{ valid } |\mathcal{R})$$

$$\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}}(\mathcal{R}) = P_{\mathtt{A_{ptxt}},\mathcal{M}\setminus\mathcal{R},}(\tilde{m} \text{ valid }, \tilde{s} \notin \mathcal{R}|\mathcal{R})$$

While the two adversaries have access to the same knowledge set obtained through the attacks, they have different success criterion. Each adversary A_{ctxt} (or A_{ptxt}) may also use different strategies in choosing $\tilde{m}$.

Definition 5. *The* **advantages** $\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntC-ATT}}(\mathcal{R})$ *and* $\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}}(\mathcal{R})$ *of an adversary with the knowledge set* $\mathcal{R}$ *obtained through the attack types ATT and constructed as above is defined as*

$$\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntC-ATT}} = \max_{A_{ctxt}} \mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntC-ATT}}(\mathcal{R})$$

$$\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}} = \max_{A_{ptxt}} \mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}}(\mathcal{R})$$

Theorem 1. `IntC` *and* `IntP` *are related through two results:*

1. `IntC` $\Rightarrow$ `IntP`
2. *In A–codes without splitting,* `IntC` $\Leftrightarrow$ `IntP`.

Proof. (sketch) For 1, note that for any adversary A_{ptxt} with knowledge set $\mathcal{R}$ who succeeds, the corresponding A_{ctxt}, using the the same strategy, succeeds too. So, $\mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}}(\mathcal{R}) \leq \mathbf{Adv}_{\mathtt{A_S}}^{\mathtt{IntP-ATT}}(\mathcal{R})$.

For 2, note that in A–codes without splitting a plaintext uniquely determines a ciphertext. Thus any $\tilde{s} \notin \mathcal{R}$ must correspond to $\tilde{m} \notin \mathcal{R}$. It follows that for every A_{ctxt} who succeeds, there is a corresponding adversary A_{ptxt} who succeeds with the same strategy. That is $\mathbf{Adv}^{\mathtt{IntP-ATT}}_{\mathtt{A_S}}(\mathcal{R}) = \mathbf{Adv}^{\mathtt{IntP-ATT}}_{\mathtt{A_S}}(\mathcal{R})$. It follows that $\mathbf{Adv}^{\mathtt{IntC-ATT}}_{\mathtt{A_S}} = \mathbf{Adv}^{\mathtt{IntP-ATT}}_{\mathtt{A_S}}$. □

Theorem 1, part 1, matches the result in [2–Thm. 3.1]. Theorem 1 shows that for A–codes without splitting, that is deterministic A–codes, we need not consider `IntP` as it is equivalent to `IntC`. In the rest of this section we only consider A–codes without splitting.

We now define ϵ–perfect and perfect ($\epsilon = 0$) `IntC`.

Definition 6. *An A–code without splitting and with secrecy (or an US AE system) provides ϵ–perfect ciphertext integrity under an attack ATT if it satisfies*

$$\mathbf{Adv}^{\mathtt{IntC-ATT}}_{\mathtt{A_S}} = \max_{m \in \mathcal{M}} \mathbf{Adv}^{\mathtt{IntC-ATT}}_{\mathtt{A_S}}(m; \mathcal{R})$$

$$\left| \mathbf{Adv}^{\mathtt{IntC-ATT}}_{\mathtt{A_S}} - \frac{|\mathcal{S}| - t_s}{|\mathcal{M}| - t_m} \right| \leq \epsilon \ .$$

where t_s is the number of plaintexts whose associated ciphertext is known through interaction with AE or VeD oracles, and t_m is the number of ciphertexts identified as valid under the active key.

This means, in particular, that when there are no observed messages, the maximum advantage of an adversary (highest probability of success) in breaking the integrity of the ciphertext is $|\mathcal{S}|/|\mathcal{M}|$. For a system with COA_{L_1}, CPA_{L_2} and CCA_{L_3} we have $t_s \leq L_1 + L_2 + L_3$ and $t_m \leq L_1 + L_2 + L_3$. In CCA and CPA both plaintext and ciphertext will be known except for the cases when the query to the VeD–oracle results in $\perp$.

In unconditional security providing protection against integrity requires redundancy, so there are more ciphertexts than plaintext.

We would like to consider results analogous to those obtained in computational security, for example theorem 3.2 of [2], relating to obtaining PS–CCA by having `IntC`∧PS–CPA.

3.3 Relationship with Classical *A*–Codes

Here we consider the relationship between the security notion `IntC`–COA_L and the notions of security considered in traditional A–codes.

Impersonation and substitution attacks in A–codes correspond to attempts to produce an acceptable message after observing 0 or 1 messages, respectively. These generalise to `IntC`–COA_0 and `IntC`–COA_1 respectively. Stinson [25] gave results for codes providing authentication and secrecy, in particular for systems satisfying the notions PS–COA_t∧`IntC`–COA_{t-1} or PS–COA_t∧`IntC`–COA_t. See section 4.

We use a well–known A–code, the polynomial A–code of [6], to illustrate view and advantage for authentication codes. The A–code is defined by $f(x) = a + bx$,

where $(a, b) \in F_q^2$ is the key, and the tag for the source state $s \in F_q$ is $f(s)$. The code satisfies $P_I = P_S = 1/q$, that is the probabilities of impersonation or substitution attacks succeeding is $1/q$. Consider the case $s \in F_3$, so we have an authentication table

(a,b)	0 1 2	(a,b)	0 1 2	(a,b)	0 1 2
$(0,0)$	0 0 0	$(0,1)$	0 1 2	$(0,2)$	0 2 1
$(1,0)$	1 1 1	$(1,1)$	1 2 0	$(1,2)$	1 0 2
$(2,0)$	2 2 2	$(2,1)$	2 0 1	$(2,2)$	2 1 0

.

In the table the rows, with (a, b), label the keys and the columns are labelled by source states. A particular entry is the tag for the source state under the key. To apply our definitions we interpret the plaintext and tag as the ciphertext, i.e. $m = (s, f(s))$. Thus there are nine different ciphertexts, $(i, j), i, j \in F_3$, rather than 3 different authentication tags. We assume the keys are chosen with equal probability.

Consider impersonation, with an empty knowledge set $\mathcal{R}$. The probability $P_{\mathtt{A},\mathcal{M}}(m) = 1/3, \forall m \in \mathcal{M}$ as required, since each message is valid under three of the nine keys and the keys are equally likely to be chosen.

Consider substitution or `IntC`–COA_1. Consider the adversary observing $(0, 0)$. The possible valid keys are now $(0, 0), (0, 1)$ and $(0, 2)$, while the ciphertexts of interest are $(1, 0)$, $(1, 1)$, $(1, 2)$, $(2, 0)$, $(2, 1)$ and $(2, 2)$. Under any particular key only 2 messages are valid, and so the probability of substitution is also $1/3$. In terms of the formula, $P_{\mathtt{A},\mathcal{M}}(m'|m) = 1/3$.

3.4 Partial Information (Side–Channel) Oracles and Attacks

As mentioned in section 2 we allow side–channels which access some, or parts of, the component algorithms. We refer to them as partial information channels, since they obtain only part of the information output by a system algorithm. Consider one such example;

Verification Oracles (V–Oracles): [16, 20] implement the verification oracle with a particular verifier's key. On input m, the V–oracle generates a result from $\{\top, \bot\}$. The queries to this oracle are called *V–queries*.

One can consider more oracles if access to individual components of the AE scheme is available. For example, one may have an oracle which takes a plaintext s and returns the first component of a ciphertext pair $m = (c_1, c_2)$, such as an oracle holding the `EncGen` algorithm for an Encrypt–then–MAC composition construction. The V–oracle is particularly useful since the veracity of a ciphertext may be made public anyway. For integrity attacks it is often enough to consider existential attacks where acceptance can be measured against a V–oracle without needing to specify a particular target plaintext. That is, if the V–oracle returns $\top$ for a fradulent ciphertext, the adversary succeeds. We see the V–oracle is the same as the VeD^*–oracle.

The availability of this oracle means we can consider a new attack.

Chosen ciphertext verification (CCVA$_L$): The adversary accesses a V–oracle and submits L ciphertexts to it. For each query m_i the oracle responds with $\top$ or $\bot$, the former if there exists an $s \in \mathcal{S}$ such that (s, m_i) is a valid pair, the latter if no such pair exists.

Attacks with V–queries can either increase, if they return $\bot$, or decrease, if they return $\top$, the chance of attacks against integrity succeeding.

4 Bounds on Authenticated Encryption Codes

An important aspect of modelling in an US environment is the possibility of obtaining bounds of two types: *Information theoretic bounds* on the success probability of attacks in terms of information theoretic measures, and *combinatorial bounds* on the relative sizes of the system spaces. In this section we summarise existing bounds on AE codes, and give some new bounds.

Stinson [24] showed the key bound for (L–fold PS)–COA$_L$ systems [12, 25], also holds for systems satisfying IntC–COA$_{L-1}$.

Theorem 2. *[24–Thm. 2] If a system is (L–fold PS)–COA_L∧IntC–COA_{L-1} then* $|\mathcal{K}| \geq \begin{pmatrix} |\mathcal{S}| \\ L \end{pmatrix}$.

More realistically one expects that if an adversary has COA$_L$ queries for attacking confidentiality they will use them against integrity too. Stinson [25] gives a bound for this system too.

Theorem 3. *[25–Thm. 4.1] If a system is (L–fold PS)–COA_L∧IntC–COA_L then* $|\mathcal{K}| \geq \begin{pmatrix} |\mathcal{S}| \\ L \end{pmatrix} \frac{|\mathcal{M}|-L}{|\mathcal{S}|-L}$.

Let P_{ATT}^{GOAL} be the probability of an adversary succeeding in the GOAL given the information under the attack model ATT. One then has

Theorem 4. *[3] If a system is (L–fold PS)–COA_L∧IntC–COA_{L-1} then*

$$|\mathcal{K}| \geq \frac{|\mathcal{S}|!}{(|\mathcal{S}|-L)!} \prod_{i=0}^{L-1} \frac{1}{P_{COA_L}^{\mathtt{IntC}}} \ . \tag{1}$$

It was also shown [14, 18] that the probabilities above satisfy

$$P_{COA_L}^{\mathtt{IntC}} \geq \frac{|\mathcal{S}|-L}{|\mathcal{M}|-L} \qquad P_{COA_L}^{\mathtt{IntC}} \geq 2^{H(\mathcal{E}|\mathcal{M}^{L+1})-H(\mathcal{E}|\mathcal{M}^{L})} \ .$$

Exploration of information theoretic bounds for more complex systems will be left to later work. Equation (1) motivates the following theorems, which we do not have space to explore here.

Theorem 5. *If a system is (L–fold PS)–CPA_L∧IntC–CPA_{L-1} then*

$$|\mathcal{K}| \geq (|\mathcal{S}|-L)\frac{|\mathcal{M}|!}{(|\mathcal{M}|-L)!} L! \prod_{i=0}^{L-1} \frac{1}{P_{CPA_L}^{\mathtt{IntC}}} \ .$$

Theorem 6. *If a system is (L–fold PS)–$CCA_L \wedge$* $\texttt{IntC}$*–CCA_{L-1} then*

$$|\mathcal{K}| \geq (|\mathcal{S}| - L)\frac{|\mathcal{M}|!L!}{(|\mathcal{S}| - L)(|\mathcal{M}| - |\mathcal{S}|)!}\prod_{i=0}^{L-1}\frac{1}{P_{CCA_L}^{\texttt{IntC}}} \ .$$

There are several open problems regarding bounds. We would like to obtain bounds for relatively simple schemes as PS–$\text{CCA}_L \wedge$ $\texttt{IntC}$–CCA_L, as well as more complicated ones where multiple queries to different types of oracles are allowed. We would also like to relate the bounds on keysizes of general composite systems to the bounds on keysize for component systems. The composite keysize for any given construction is directly related to the components and the composition method.

5 Construction Composition

The form of $\texttt{AuthEnc}$ *may* follow one of three composition methods, described by [2] for the symmetric key setting, [1] for the public key setting and applied in [21] to the asymmetric US setting. These assume one has an encryption scheme $\mathcal{E}$, with encryption algorithm $\texttt{EncGen}$ and a MAC (authentication scheme) $\mathcal{A}$, with authentication algorithm $\texttt{MACGen}$.

- Encrypt–and–MAC: For a plaintext s one generates the ciphertext to be transmitted as the pair $(\texttt{EncGen}(s), \texttt{MACGen}(s))$. That is, the component schemes are independently applied to the plaintext.
- MAC–then–encrypt: For a plaintext s one generates the ciphertext to be transmitted as the pair $(\texttt{EncGen}(s), \texttt{EncGen}(\texttt{MACGen}(s)))$. That is, the plaintext is tagged and then both parts are encrypted.
- Encrypt–then–MAC: For a plaintext s one generates the ciphertext to be transmitted as the pair $(\texttt{EncGen}(s), \texttt{MACGen}(\texttt{EncGen}(s)))$. That is, the plaintext is encrypted and then MAC'd.

Shikata *et al* [21] consider only the third to be always "secure" as a composition method if the components are "secure". In the asymmetric setting, including the public key setting, the MAC component is replaced by a signature component.

6 A Direct Construction for Authenticated Encryption

Composition is a standard method for constructing authentication codes with secrecy, but not the only method. In this section we use a simple example to illustrate "direct" US AE. In this scheme the verified decryption process cannot be broken into independent verification and decryption processes, a returned plaintext is always valid.

Let $\mathcal{S} = \{0, 1\}$ be the set of plaintexts, and $\mathcal{M} = \{0, 1, 2, 3\}$ the set of ciphertexts. The algorithms for the AE are:

1. `KeyInt:` The trusted initialiser chooses 2 distinct elements $\{a_0, a_1\}$ of $\mathcal{M}$. The elements a_i are sent to Alice and Bob.
2. `AuthEnc:` For a chosen plaintext $s \in \mathcal{S}$ Alice broadcasts a_s.
3. `VerDec:` Upon receiving a_s Bob finds the value of s for which $a'_s{=}a_s$, and accepts s as authentic. If no such s exists, m is rejected.

There are 12 keys, corresponding to pairs of distinct elements from $\mathcal{M}$. We assume the keys are all equally likely to be used, and furthermore that the plaintexts are equally likely to be used.

Without observing a ciphertext the probability of any (s, m) pair being valid is 1/4. The probability of impersonation is 1/2, since out of the four messages two are valid under a key. Consider that Oscar observes a valid ciphertext m. The probability of Oscar generating another valid ciphertext is now 1/3, since only one of the unobserved ciphertexts corresponds to a plaintext. The probability of substitution is therefore lower than the probability of impersonation, unlike in Cartesian A–codes.

For confidentiality we consider the probability of Oscar associating an observed ciphertext m with the correct plaintext s. The probability is 1/2, and is so even if both valid ciphertexts are observed. If Oscar is allowed a query which identifies a valid pair (s', m') then the association between an observed message and the correct source state is obvious. Thus this system doesn't satisfy `IntC`–CPA_1, `IntC`–CCA_1 or `IntC`–CCVA_1.

We have argued this system satisfies the notion PS–$\mathrm{COA}_2 \wedge$`IntC`–COA_1. From Equation (1) we have, for $P^{\texttt{IntC}}_{COA_0} = 1/2$ and $P^{\texttt{IntC}}_{COA_1} = 1/3$, $|\mathcal{K}| \geq 12$. This system is optimal with respect to meeting the number of keys and also, examining the other bounds in section 4, with respect to meeting the bounds on attack probabilities.

This direct scheme can be generalised, but it more interesting to note that every A–code with secrecy is of this form. The verified decryption in an A–code with secrecy can be represented by a **lookup table**. For ciphertext m and key k then column m and row k contains either the plaintext s or nothing, the latter corresponding to an invalid ciphertext. Every A–code with secrecy is equivalent to a generalisation with a subset of all permutations as the keys.

7 Summary

We have developed a framework for US AE using recent formulations of US A–codes [16] and US encryption schemes [12]. We have restricted our attention to symmetric AE in order to clarify and observe subtleties which may be lost in a direct analysis of asymmetric AE.

We have defined the goals, attack models and oracles used to build security notions. We have summarised some bounds relevant to authenticated encryption. Due to space limitations we have only analysed a simple direct construction to illustrate some issues for symmetric AE schemes.

We have given several open questions related to bounds, entropy, probabilities and composition in section 4. It is expected some of these questions will

require significant effort to solve. There is also the need to extend this work to the asymmetric case. An analysis of the relationships between the various security notions, as considered extensively in computational security [1, 2], would be useful. Applying this framework to the analysis of specific US AE [26, 27] is important also.

Acknowledgement

The authors appreciate discussions with Dr. Joosang Baek.

References

1. J. An, Y. Dodis & T. Rabin 'On the security of joint signature and encryption.' *EuroCrypt'02* LNCS **2332** (2002) 83–107.
2. M. Bellare & C. Namprempre 'Authenticated encryption: relations among notions and analysis of the generic composition paradigm.' *AsiaCrypt'00* LNCS **1976** (2000) 317–30.
3. L.R.A. Casse, K.M. Martin & P.R. Wild 'Bounds and characterizations of authentication/secrecy schemes.' *Designs, Codes and Cryptography* **13** (1998) 107–29.
4. M. De Soete 'Some constructions for authentication–secrecy codes' *Eurocrypt'88* LNCS **330** (1988) 57–75.
5. M. De Soete 'Bounds and constructions for authentication–secrecy codes with splitting.' *Crypto'88* LNCS **403** (1989) 311–7.
6. Y. Desmedt, Y. Frankel & M. Yung 'Multi–receiver/multi–sender network security: efficient authenticated multicast/feedback.' *IEEE Infocom'92* (1992) 2045–54.
7. E.N. Gilbert, F.J. MacWilliams & N.J.A. Sloane 'Codes which detect deception.' *Bell System Tech. Journal* **53** (1974) 405–24.
8. G. Hanaoka, J. Shikata, Y. Zheng & H. Imai 'Unconditionally secure digital signature schemes admitting transferability.' *Asiacrypt'00* LNCS **1976** (2000) 130–42.
9. G. Hanaoka, J. Shikata, Y. Hanaoka & H. Imai 'Unconditionally secure anonymous encryption and group authentication.' *Asiacrypt'02*, LNCS **2501** (2002) 81–99.
10. T. Johansson 'Contributions to unconditionally secure authentication.' Ph.D. Thesis, (Lund University, Sweden, 1994).
11. K. Kurosawa and S. Obana 'Characterisation of (k, n) multi–receiver authentication.' *ACISP'97* LNCS **1720** (1997) 204–15.
12. L. McAven, R. Safavi–Naini & M. Yung 'Unconditionally secure encryption under strong attacks.' *ACISP 2004* LNCS **3108** (2004) 427–439.
13. M. Naor & M. Yung 'Public–key cryptosystems provably secure against chosen–ciphertext attacks.' *22nd STOC* ACM (1990) 427–437.
14. R.S. Rees & D.R. Stinson 'Combinatorial characterizations of authentication codes.' *Designs, Codes and Cryptography* **7** (1996) 239–59.
15. R. Safavi–Naini and H. Wang 'Multireceiver authentication codes: Models, bounds, constructions and extensions.' *Information and Computation* **151** (1999) 148–72.
16. R. Safavi–Naini, L. McAven & M. Yung 'General group authentication codes and their relation to "Unconditionally secure signatures".' *Public Key Cryptography* (2004) 231–47.
17. G.J. Simmons 'A Cartesian product construction for unconditionally secure authentication codes that permit arbitration' *J.Crypt.* **2**(2) (1990) 77–104.

18. A. Sgarro 'Information–theoretic bounds for authentication frauds.' *J. Computer Security* **2** (1993) 53–63.
19. C. E. Shannon 'Communication theory of secrecy systems.' *Bell System Tech. Journal* **28** (1949) 269–79.
20. J. Shikata, G. Hanaoka, Y. Zheng & H. Imai 'Security notions for unconditionally secure signature schemes' *Eurocrypt'02* LNCS **2332** (2002) 434–49.
21. J. Shikata, G. Hanaoka, Y. Zheng, T. Matsumoto & H. Imai 'Unconditionally secure authenticated encryption.' *IEICE Trans. Fundamentals* **E87–A**(5) (2004).
22. G.J. Simmons 'Authentication theory/coding theory.' *Crypto'84* LNCS **196** (1984) 411–32.
23. B. Smetts, P. Vanroose & Z.X. Wan 'On the construction of authentication codes with secrecy and codes withstanding spoofing attacks of order $L \geq 2$.' (1988) 306–.
24. D. R. Stinson 'Some constructions and bounds for authentication codes.' *J.Crypt.* **1** (1988) 37–51.
25. D. R. Stinson 'The combinatorics of authentication and secrecy codes.' *J.Crypt.* **2** (1990) 23–49.
26. X. Tian & C. Ding 'A construction of authentication codes with secrecy.' *Progress in Computer Science and Applied Logic* **23** 319–30.
27. T. V. Trung 'On the construction of authentication and secrecy codes.' *Designs, Codes and Cryptography* **5** (1995) 269–80.

Faster Variants of the MESH Block Ciphers

Jorge Nakahara Júnior

jorge_nakahara@yahoo.com.br

Abstract. This paper describes new variants of the MESH block ciphers, that use the same group operations of the IDEA cipher but operate on 8-bit words, and are estimated to be faster than the AES on 8-bit processors. These results corroborate the high flexibility of the MESH cipher design, demonstrating their high adaptability to 8-bit smart card processors, such as the 8051. All the design features of the original 16-bit word-oriented MESH ciphers were preserved for the 8-bit variants: complete diffusion in a single round; MA-boxes alternating modular addition and multiplication; asymmetric key-mixing layers for odd and even rounds; new key schedule algorithms with fast key avalanche; the same computational framework holds for encryption and decryption. Preliminary Square and algebraic attack results are described, and resistance to other modern cryptanalytic techniques is also expected.

Keywords: byte-oriented block ciphers, IDEA, smart cards, algebraic attacks.

1 Introduction

The IDEA block cipher [18] dates back to 1991, and evolved from the PES cipher with improved resistance against differential cryptanalysis [19]. IDEA has withstood several cryptanalytic attacks ever since: an attack by Meier in [20] on up to 2.5 rounds; a differential-linear attack by Borst *et al.* [7] on 3 rounds; a truncated-differential attack by Borst *et al.* [7] on 3.5 rounds; a square attack by Demirci [14] on up to 4 rounds; an impossible differential attack by Biham *et al.* [2] on up to 4.5-rounds; and finally, a meet-in-the-middle attack by Demirci *et al.* [15] on 5 rounds. There are attacks on the full 8.5-round IDEA, such as linear cryptanalysis [11, 26], differential-linear [17], and boomerang attacks [4], but all of them require the assumption of weak keys, namely, user keys that cause multiplicative subkeys with particular values.

IDEA became well-known probably because it was embedded in the widely distributed Pretty Good Privacy software package from Phil Zimmerman [16], for general purpose file encryption. Curiously, for more than 12 years since the announcement of IDEA, no IDEA nor PES variant has ever been suggested with a block size larger than 64 bits, in any worldwide event such as the AES [1], NESSIE [24] or CRYPTREC [8]. Such variant(s) would certainly be a useful primitive for the construction of hash functions and MACs [21].

The original MESH ciphers exploited the high flexibility of the MA-box (Multiplication-Addition box) of IDEA [18], in order to operate on larger text blocks.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 162–174, 2004.

All of these ciphers employ three group operations on 16-bit words: exclusive-or, addition in $\mathbb{Z}_{2^{16}}$, and multiplication in $\mathrm{GF}(2^{16}+1)$, with the exception that $0 \equiv 2^{16}$. The MA-box is a bijective mapping composed of addition and multiplication operations alternated and computed in a zig-zag order. Fig. 1 shows an example of the MA-box.

Previous analyses of 16-bit word-oriented MESH ciphers [22] showed a relatively high margin of security against modern cryptanalytic techniques but the software performance was estimated to be much lower than that of the AES [1]. This paper explores one further aspect of the MESH ciphers, namely the word size can be reduced from 16 bits to 8 bits, allowing all internal operations to be efficiently performed on low-cost 8-bit smart cards. These 8-bit word MESH variants are denoted with the "(8)" sufix to distinguish them from the original 16-bit word-oriented MESH ciphers. These 8-bit word MESH variants allow variable block sizes in increments of 16 bits.

This paper is organized as follows: Sect. 2 describes MESH-64(8); Sect. 3 describes MESH-128(8); Sect. 4 describes Square attacks on reduced-round MESH variants; Sect. 5 describes preliminary results of algebraic attacks on the MESH variants; Sect. 6 presents estimates of software performance of the MESH variants, and compares them to the AES; Sect. 7 concludes the paper.

2 The MESH-64(8) Block Cipher

MESH-64(8) is an iterated block cipher that operates on 64-bit blocks, uses a 128-bit key, and consists of eight rounds plus an output transformation. MESH-64(8) employs the same three group operations of IDEA, but all internal operations are on 8-bit words: exclusive-or, denoted $\oplus$; addition in $\mathbb{Z}_{2^8}$, denoted $\boxplus$; and multiplication in $\mathrm{GF}(2^8+1)$, denoted $\odot$, with the exception that $0 \equiv 2^8$. The MA-box is the core of a round in MESH-64(8), and contains two layers of multiplication and addition, interleaved, and computed in a zig-zag order (Fig. 1). Each layer in the MA-box costs two additions and two multiplications. Although each additional layer might strengthen the MA-box, it also degrades the cipher performance because the operations are in series. The use of two layers in the MA-box, as in IDEA, is a balance between performance and security. The round structure of MESH-64(8) consists of two halves: a subkey mixing[1] and an MA half-round (Fig. 1). The number of rounds can be increased beyond eight, but must be set to an even number to allow encryption and decryption to use the same computational graph. This is a consequence of the asymmetric key mixing for odd and even rounds. Let $X^{(i)} = (X_1^{(i)}, X_2^{(i)}, X_3^{(i)}, X_4^{(i)}, X_5^{(i)}, X_6^{(i)}, X_7^{(i)}, X_8^{(i)})$ be the input to the i-th round.

[1] Subkeys are denoted $Z_i^{(j)}$, where the superscript j is surrounded by parenthesis to avoid confusion with the power operation.

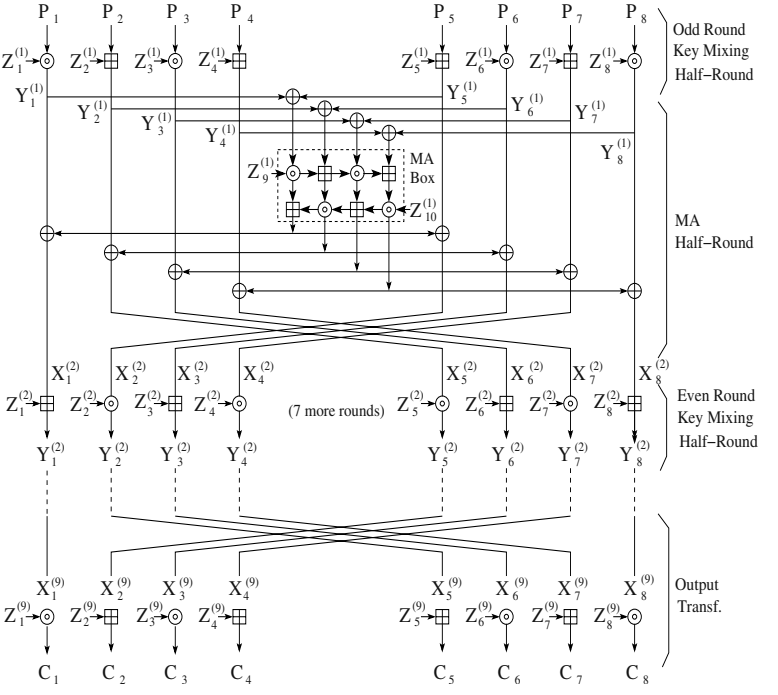

Fig. 1. Computational graph of MESH-64(8)

Then, the output of the i-th key mixing is the $Y^{(i)}$ tuple, for $1 \leq i \leq 8$:

$$Y^{(i)} = \begin{cases} (X_1^{(i)} \odot Z_1^{(i)}, X_2^{(i)} \boxplus Z_2^{(i)}, X_3^{(i)} \odot Z_3^{(i)}, X_4^{(i)} \boxplus Z_4^{(i)}, \\ X_5^{(i)} \boxplus Z_5^{(i)}, X_6^{(i)} \odot Z_6^{(i)}, X_7^{(i)} \boxplus Z_7^{(i)}, X_8^{(i)} \odot Z_8^{(i)}), \text{ for } i \text{ odd} \\ (X_1^{(i)} \boxplus Z_1^{(i)}, X_2^{(i)} \odot Z_2^{(i)}, X_3^{(i)} \boxplus Z_3^{(i)}, X_4^{(i)} \odot Z_4^{(i)}, \\ X_5^{(i)} \odot Z_5^{(i)}, X_6^{(i)} \boxplus Z_6^{(i)}, X_7^{(i)} \odot Z_7^{(i)}, X_8^{(i)} \boxplus Z_8^{(i)}), \text{ for } i \text{ even} \end{cases}$$

The $Y^{(i)}$ tuple is the input to the i-th MA half-round. The i-th MA half-round output is $X^{(i+1)} = (Y_1^{(i)} \oplus W_1^{(i)}, Y_5^{(i)} \oplus W_1^{(i)}, Y_6^{(i)} \oplus W_2^{(i)}, Y_7^{(i)} \oplus W_3^{(i)}, Y_2^{(i)} \oplus W_2^{(i)}, Y_3^{(i)} \oplus W_3^{(i)}, Y_4^{(i)} \oplus W_4^{(i)}, Y_8^{(i)} \oplus W_4^{(i)})$, where:

$$\mathbf{W_4^{(i)}} = (((Y_1^{(i)} \oplus Y_5^{(i)}) \odot Z_9^{(i)} \boxplus (Y_2^{(i)} \oplus Y_6^{(i)})) \odot (Y_3^{(i)} \oplus Y_7^{(i)}) \boxplus (Y_4^{(i)} \oplus Y_8^{(i)})) \odot Z_{10}^{(i)},$$

$$\mathbf{W_3^{(i)}} = W_4^{(i)} \boxplus ((Y_1^{(i)} \oplus Y_5^{(i)}) \odot Z_9^{(i)} \boxplus (Y_2^{(i)} \oplus Y_6^{(i)})) \odot (Y_3^{(i)} \oplus Y_7^{(i)}),$$

$$\mathbf{W_2^{(i)}} = W_3^{(i)} \odot ((Y_1^{(i)} \oplus Y_5^{(i)}) \odot Z_9^{(i)} \boxplus (Y_2^{(i)} \oplus Y_6^{(i)})),$$

and

$$\mathbf{W_1^{(i)}} = W_2^{(i)} \boxplus (Y_1^{(i)} \oplus Y_5^{(i)}) \odot Z_9^{(i)}.$$

The last half-round, or the output transformation (OT), consists of a fixed involution of the words in a block, followed by a key mixing half-round.

2.1 The Key Schedule of MESH-64(8)

The key schedule for MESH-64(8) is defined as follows:

- 8-bit constants c_i are defined by $c_0 = 1$, and $c_i = 2 \cdot c_{i-1}$ for $i \geq 1$, with multiplication in the field GF(2^8) represented as GF(2)[x]/$p(x)$, where $p(x) = x^8 + x^4 + x^3 + x + 1$ is a primitive polynomial in GF(2)[x]. The constant '2' is represented by the polynomial 'x'.
- The 128-bit key is partitioned into sixteen 8-bit words K_i, $0 \leq i \leq 15$. The first sixteen subkeys are assigned to $Z^{(1)}_{i+1} = K_i \oplus c_i$, $0 \leq i \leq 9$, and $Z^{(2)}_{j \bmod 10+1} = K_j \oplus c_j$, $10 \leq j \leq 15$.
- Each subsequent 8-bit subkey is computed as follows:[2]

$$Z^{(h(i))}_{l(i)} = \Big(\Big(\Big(Z^{(h(i-16))}_{l(i-16)} \boxplus Z^{(h(i-12))}_{l(i-12)}\Big) \oplus Z^{(h(i-3))}_{l(i-3)}\Big) \boxplus Z^{(h(i-2))}_{l(i-2)}\Big) \lll 1 \oplus c_i\,, \tag{1}$$

  for $16 \leq i \leq 87$, where '$\lll 1$' means one-bit left rotation, $h(i) = i \operatorname{div} 10 + 1$, and $l(i) = i \bmod 10 + 1$.

The key schedule of MESH-64(8) achieves fast avalanche because the $h(i)$ and $l(i)$ indices in (1) are based on the primitive polynomial $r(x) = x^{16} + x^{14} + x^{13} + x^4 + 1$, and the interleaving of $\boxplus$ and $\oplus$. For example, each subkey starting with $Z^{(3)}_2$ already depends upon all sixteen user key words. The dependence of (1) on $r(x)$ can be made clear by ignoring the left rotation for a while, changing the $\boxplus$ to $\oplus$, denoting $Z^{(h(i))}_{l(i)}$ simply as $Z^{(i)}$, and adding 16 to the indices: $Z^{(i+16)} = Z^{(i)} \oplus Z^{(i+4)} \oplus Z^{(i+13)} \oplus Z^{(i+14)} \oplus c_i$. The constants c_i are used to avoid patterns in the subkeys. Without these constants, the key with 128 zero bits would result in all subkeys being equal to zero.

Decryption in MESH-64(8) uses the same framework in Fig. 1 as for encryption, but with transformed round subkeys. If the encryption subkeys for the r-th round are denoted $(Z^{(r)}_1, \ldots, Z^{(r)}_{10})$, for $1 \leq r \leq 8$, and $(Z^{(9)}_1, \ldots, Z^{(8)}_8)$ for the output transformation, then the decryption round subkeys are:

- $((Z^{(9)}_1)^{-1}, -Z^{(9)}_2, (Z^{(9)}_3)^{-1}, -Z^{(9)}_4, -Z^{(9)}_5, (Z^{(9)}_6)^{-1}, -Z^{(9)}_7, (Z^{(9)}_8)^{-1}, Z^{(8)}_9, Z^{(8)}_{10})$, for the first decryption round.
- $(-Z^{(10-r)}_1, (Z^{(10-r)}_5)^{-1}, -Z^{(10-r)}_6, (Z^{(10-r)}_7)^{-1}, (Z^{(10-r)}_2)^{-1}, -Z^{(10-r)}_3, (Z^{(10-r)}_4)^{-1}, -Z^{(10-r)}_8, Z^{(9-r)}_9, Z^{(9-r)}_{10})$, for the r-th even round, $r \in \{2, 4, 6, 8\}$.
- $((Z^{(10-r)}_1)^{-1}, -Z^{(10-r)}_5, (Z^{(10-r)}_6)^{-1}, -Z^{(10-r)}_7, -Z^{(10-r)}_2, (Z^{(10-r)}_3)^{-1}, -Z^{(10-r)}_4, (Z^{(10-r)}_8)^{-1}, Z^{(9-r)}_9, Z^{(9-r)}_{10})$, for the r-th odd round, $r \in \{3, 5, 7\}$.

[2] Bit rotation has higher precedence than $\oplus$.

. $((Z_1^{(1)})^{-1}, -Z_2^{(1)}, (Z_3^{(1)})^{-1}, -Z_4^{(1)}, -Z_5^{(1)}, (Z_6^{(1)})^{-1}, -Z_7^{(1)}, (Z_8^{(1)})^{-1})$ for the output transformation.

3 The MESH-128(8) Block Cipher

MESH-128(8) is an iterated block cipher that operates on 128-bit blocks, uses a 256-bit key, and consists of eight rounds plus an output transformation. MESH-128(8) employs the same three group operations of MESH-64(8). Similar to MESH-64(8) and IDEA, no two consecutive operations are the same. The round structure of MESH-128(8) consists of two halves: a key mixing and an MA half-round (Fig. 2). The computation of intermediate steps in the MA-box follows a similar procedure as for MESH-64(8).

The last half-round, or the output transformation, consists of a fixed involution of the words in a block, followed by a key mixing half-round.

3.1 The Key Schedule of MESH-128(8)

The key schedule for MESH-128(8) is defined as follows:

- The same 8-bit constants c_i defined in MESH-64(8) are used in MESH-128(8).
- The 256-bit key is partitioned into 32 8-bit words K_i, $0 \le i \le 31$, which are assigned to the first 32 subkeys: $Z_{i+1}^{(1)} = K_i \oplus c_i$, $0 \le i \le 17$, and $Z_{j \bmod 18+1}^{(2)} = K_j \oplus c_j$, $18 \le j \le 31$.
- Each subsequent 8-bit subkey is computed as follows:

$$Z_{l(i)}^{(h(i))} = \Big(\Big(\Big(\Big(\Big(Z_{l(i-32)}^{(h(i-32))} \boxplus Z_{l(i-31)}^{(h(i-31))}\Big) \oplus Z_{l(i-30)}^{(h(i-30))}\Big) \boxplus Z_{l(i-29)}^{(h(i-29))}\Big) \oplus Z_{l(i-27)}^{(h(i-27))}\Big) \boxplus Z_{l(i-25)}^{(h(i-25))}\Big) \lll 1 \oplus c_i\,, \tag{2}$$

  for $31 \le i \le 159$, where '$\lll 1$' means one-bit left rotation, $h(i) = i \text{ div } 18 + 1$, and $l(i) = i \bmod 18 + 1$.

The key schedule of MESH-128(8) achieves fast avalanche because the $h(i)$ and $l(i)$ indices in (2) are based on the primitive polynomial $q(x) = x^{32} + x^7 + x^5 + x^3 + x^2 + x + 1$, and the interleaving of $\boxplus$ and $\oplus$ (similar to MESH-64(8)). For example, each subkey from $Z_3^{(4)}$ on, already depends upon all 32 key words.

Decryption in MESH-128(8) uses the same framework in Fig. 2 as for encryption, but with transformed round subkeys. If the encryption subkeys for the r-th round are denoted $(Z_1^{(r)}, \ldots, Z_{18}^{(r)})$, for $1 \le r \le 8$, and $(Z_1^{(9)}, \ldots, Z_{16}^{(9)})$ for the output transformation, then the decryption round subkeys are:

. $((Z_1^{(9)})^{-1}, -Z_2^{(9)}, (Z_3^{(9)})^{-1}, -Z_4^{(9)}, (Z_5^{(9)})^{-1}, -Z_6^{(9)}, (Z_7^{(9)})^{-1}, -Z_8^{(9)}, -Z_9^{(9)}, (Z_{10}^{(9)})^{-1}, -Z_{11}^{(9)}, (Z_{12}^{(9)})^{-1}, -Z_{13}^{(9)}, (Z_{14}^{(9)})^{-1}, -Z_{15}^{(9)}, (Z_{16}^{(9)})^{-1}, Z_{17}^{(8)}, Z_{18}^{(8)})$, for the first decryption round.

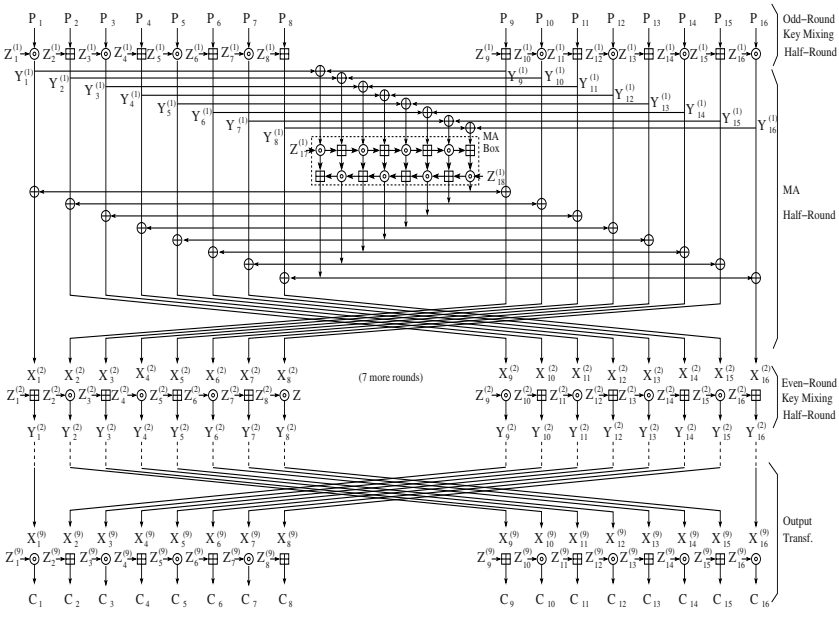

Fig. 2. Computational graph of MESH-128(8)

- $(-Z_1^{(10-r)}, (Z_9^{(10-r)})^{-1}, -Z_{10}^{(10-r)}, (Z_{11}^{(10-r)})^{-1}, -Z_{12}^{(10-r)}, (Z_{13}^{(10-r)})^{-1}, -Z_{14}^{(10-r)}, (Z_{15}^{(10-r)})^{-1}, (Z_2^{(10-r)})^{-1}, -Z_3^{(10-r)}, (Z_4^{(10-r)})^{-1}, -Z_5^{(10-r)}, (Z_6^{(10-r)})^{-1}, -Z_7^{(10-r)}, (Z_8^{(10-r)})^{-1}, -Z_{16}^{(10-r)}, Z_{17}^{(9-r)}, Z_{18}^{(9-r)})$, for the r-th even round, $r \in \{2, 4, 6, 8\}$.
- $((Z_1^{(10-r)})^{-1}, -Z_9^{(10-r)}, (Z_{10}^{(10-r)})^{-1}, -Z_{11}^{(10-r)}, (Z_{12}^{(10-r)})^{-1}, -Z_{13}^{(10-r)}, (Z_{14}^{(10-r)})^{-1}, -Z_{15}^{(10-r)}, -Z_2^{(10-r)}, (Z_3^{(10-r)})^{-1}, -Z_4^{(10-r)}, (Z_5^{(10-r)})^{-1}, -Z_6^{(10-r)}, (Z_7^{(10-r)})^{-1}, -Z_8^{(10-r)}, (Z_{16}^{(10-r)})^{-1}, Z_{17}^{(9-r)}, Z_{18}^{(9-r)})$, for the r-th odd round, $r \in \{3, 5, 7\}$.
- $((Z_1^{(1)})^{-1}, -Z_2^{(1)}, (Z_3^{(1)})^{-1}, -Z_4^{(1)}, (Z_5^{(1)})^{-1}, -Z_6^{(1)}, (Z_7^{(1)})^{-1}, -Z_8^{(1)}, -Z_9^{(1)}, (Z_{10}^{(1)})^{-1}, -Z_{11}^{(1)}, (Z_{12}^{(1)})^{-1}, -Z_{13}^{(1)}, (Z_{14}^{(1)})^{-1}, -Z_{15}^{(1)}, (Z_{16}^{(1)})^{-1})$, for the output transformation.

The suggested minimum number of rounds in MESH-128(8) was 8.5 because it represents a balance between security and efficiency (similarly for the number of layers in the MA-box).

4 A Square Attack on the MESH Variants

The original Square attack was developed by Knudsen *et al.* [12] as a dedicated attack against the Square block cipher, but this attack can be applied similarly to other word-oriented block ciphers. Since all internal operations in the MESH

variants operate on 8-bit words, Square attacks employ preferably this word size. Recall that Square attacks and its variants are the currently best known attacks on reduced-round versions of the AES cipher [13]. The terminology for this attack follows [12]. The design of the MA-box of the MESH variants imply that the most effective λ-set that can propagate across the MA-box, has the form $(PPPA)$, and causes the output λ-set [3] $(AAAA)$. The reason is that this λ-set preserves the highest number of active/passive words across the MA-box, among all possible input λ-sets. This λ-set for the MA-box translates into the following one-round λ-sets: $(P\ P\ P\ P\ P\ P\ P\ A) \rightarrow (A\ A\ A\ A\ A\ A\ A\ ?)$, and $(P\ P\ P\ A\ P\ P\ P\ P) \rightarrow (A\ A\ A\ A\ A\ A\ ?\ A)$ which can be used to attack 2-round MESH-64(8). Without loss of generality, an attack on the first two rounds of MESH-64(8) is as follows:

- choose 256 plaintexts with words according to the λ-set $(PPPPPPPA)$; the λ-set after 1.5 rounds is $(AAAAAAA?)$, but after 2 rounds the λ-set becomes (????????);
- guess the subkeys $Z_9^{(2)}$, $Z_{10}^{(2)}$, and decrypt the last MA half-round for all texts in the output λ-set;
- keep the subkey for which the decrypted λ-set has the form $(A\ A\ A\ A\ A\ A\ A\ ?)$; this is the distinguisher, and each active word is used to test for the correct key guess;
- for a wrong subkey pair there is a chance of 2^{-8} that each of the first seven words is active, each of which works as a distinguisher. Therefore, there is a chance of 2^{-56} that any wrong subkey produces all seven active words in this λ-set. Since there are $2^{16} - 1$ wrong subkeys, the expected number of wrong subkeys that survive this filtering is $2^{-56} \cdot (2^{16} - 1) < 1$, and thus only the correct subkey is expected to survive.

Therefore, the attack requires 2^8 chosen plaintexts, and time equivalent to about $2^8 \cdot 2^{16} = 2^{24}$ half-round decryptions, or $\frac{1}{4} \cdot 2^{24} = 2^{22}$ 2-round computations.

Using the same distinguisher, but extending the attack to 2.5 rounds, one can guess the 10 subkeys of the last two half-rounds, or 80 subkey bits. With two λ-sets, the expected number of false alarms (wrong subkeys) that survive this filtering is $(2^{-56})^2 \cdot 2^{80} < 1$. The time complexity is $2 \cdot 2^8 \cdot 2^{80} = 2^{89}$ 1-round computations, or $\frac{1}{2.5} \cdot 2^{89} \approx 2^{88}$ 2.5-round computations.

For 3 rounds, one can use the same distinguisher, but recover 18 subkeys from the last 1.5 rounds, or 144 subkey bits. With three λ-sets, the expected number of wrong subkeys surviving this filtering is $(2^{-56})^3 \cdot 2^{144} < 1$. The time complexity is about $3 \cdot 2^8 \cdot 2^{144} \approx 2^{153.5}$ 1.5-round computations, or $2^{152.5}$ 3-round computations, which is more than the effort of an exhaustive key search.

The attack on 2-round MESH-128(8) has the same complexity as on MESH-64(8), but the atack on 2.5 rounds recovers 18 subkeys, or 144 subkey bits, and requires three λ-sets, at the cost of $3 \cdot 2^8 \cdot 2^{144} \cdot \frac{1}{2.5} \approx 2^{152}$ 2.5-round encryptions. The attack on 3 rounds recovers 20 subkeys of the last 1.5 rounds, and also uses three λ-sets at the cost of $3 \cdot 2^8 \cdot 2^{160} \cdot \frac{1}{3} = 2^{169}$ 3-round computations.

[3] 'A' denotes an active word; 'P' denotes a passive word and '?' an unbalanced word.

5 The Courtois-Pieprzyk Attack

In [10], Courtois and Pieprzyk described a potential attack on the AES. Their approach initially derived multivariate algebraic equations (with probability one) from the AES S-box, and further obtained a system of (overdefined) multivariate equations, more specifically quadratic equations, which would be solved by algorithms called XL [9], or XSL [10], requiring a few known plaintext/ciphertext pairs.

One requirement of the algebraic attack is that quadratic equations can be derived from the S-box or other non-linear operations. For the MESH ciphers the multiplication is the main non-linear operation, and it can be represented by a combination of two S-boxes, one based on discrete exponentiation and another on discrete logarithm. Recall that $\mathrm{GF}(2^8+1)$ is a cyclic group, which means that there is a generator $g \in \mathbb{Z}^*_{2^8+1}$, such that $< g >= \mathrm{GF}(2^8+1)$. More precisely, for $\mathrm{GF}(2^8+1)$ there are $\phi(\phi(2^8+1)) = 128$ generators [21–p. 70, Chap. 2], where ϕ is Euler's totient function. Consider, for example, $g = 5$. Then, for any $x, y \in \mathbb{Z}_{2^8}$ there are $a, b \in \mathrm{GF}(2^8+1)$ such that $5^x = a$, and $5^y = b$. Analogously, a discrete logarithm function can be defined as $\log_5 a = x \Leftrightarrow 5^x = a$, for $x \in \mathbb{Z}_{2^8}$, and $a \in \mathrm{GF}(2^8+1)$. It follows that $a \odot b = 5^x \cdot 5^y = 5^{\log_5 a \boxplus \log_5 b}$. Therefore, if discrete exponentiation and logarithm tables are available (at the cost of only $2 \cdot 2^8 = 512$ bytes of memory), then a $\odot$ operation in $\mathrm{GF}(2^8+1)$ can be accomplished in constant time with three table lookups and an addition.

For the 8×8-bit exponentiation S-box, and using any generator, there are at most $\binom{16}{0} + \binom{16}{1} = 17$ linear terms, $\binom{16}{2} = 120$ quadratic terms, $\binom{16}{3} = 560$ cubic terms, and $\binom{16}{4} = 1820$ 4th-degree terms. But, there are only $2^8 = 256$ inputs. Since $256 < \binom{16}{0} + \binom{16}{1} + \binom{16}{2} + \binom{16}{3} = 697$, and $256 > \binom{16}{0} + \binom{16}{1} + \binom{16}{2} = 137$, it means that there can be at most cubic (multivariate) equations (with probability 1), involving 16 input/output bits. Several computations of potential algebraic equations for the discrete logarithmic and exponentiation tables, with all possible 128 generators, using the same approach as Biryukov and De Cannière in [3], did not detect any quadratic equation involving the eight input bits and the eight output bits of the exponentiation S-box[4] for any generator. Similarly, no cubic nor 4th-degree equations were found, implying that the discrete exponentiation and logarithmic S-boxes do not present a simple algebraic structure such as that of the AES.

Nonetheless, from Daemen et al. [11], there are particular subkey values, 0 and 1, that turn the multiplication into a much simpler operation, namely the identity and modular subtraction operations[5], respectively. Notwithstanding, the

[4] The analysis can be restricted to one S-box, since the g^x S-box is the inverse of the $\log_g$ S-box, and any algebraic equation for one of them can be converted to the other by just changing the input for the output variables.

[5] Multiplication by 1 is the identity operation; multiplication by 0, in $\mathrm{GF}(2^8+1)$, is equivalent to multiplying by -1.

MESH variants have countermeasures against these cases. Recall that in IDEA all multiplications always involve one subkey as an operand. Therefore, if the key schedule is not carefully designed, then the multiplication operations could, in principle, be weakened or manipulated, by appropriately selecting a key non-randomly (such that many subkeys end up with value 0 or 1). In the MESH variants described in this paper there are two provisions to avoid this weakness:

- not all multiplications involve a subkey directly; the MA-boxes of the MESH variants (Fig. 1 and 2) contain multiplications involving only intermediate text values; therefore, these operations become text-and-key dependent;
- the key schedule algorithms do not have the key bit overlapping property of IDEA; consequently, key reconstruction becomes harder in the case of key-recovery attacks, and it also helps avoid several weak multiplicative subkeys to happen simultaneously along the full cipher.

6 Software Performance

The software performance of the 8-bit MESH ciphers was estimated from the number of 8-bit operations in each cipher. This approach did not take into account the possible paralellisms in the cipher structure, but was used as a rough measure of computational cost. These ciphers are compared to the AES, which also perform all internal operations in 8-bit words (Table 1). The measurement was made in CPU cycles per byte encrypted so that ciphers with different block sizes can be compared. The diffusion rate in Table 1 is the number of rounds after which all ciphertext bits depend (non-linearly) on all plaintext bits.

The performance estimates assume that **each of the following operations cost only one CPU cycle**: xtime (in AES), $\oplus$, $\boxplus$, (fixed) bit rotation[6]. Taking the 8051 processor as a reference, one table lookup operation will typically cost twice an $\oplus$ and an $\boxplus$.

For AES-128, namely AES with a 128-bit key, the number of 8-bit operations is computed as follows: $16 \cdot 10 = 160$ S-box look-ups, $16 \cdot 2 \cdot 9 = 288$ xtime operations, $16 \cdot 11 + 4 \cdot 15 \cdot 9 = 716$ $\oplus$'s, or 1324 cycles per 128-bit block.

For AES-192, the number of 8-bit operations is computed as follows: $16 \cdot 12 = 192$ S-box look-ups, $16 \cdot 2 \cdot 11 = 352$ xtime operations, $16 \cdot 13 + 4 \cdot 15 \cdot 11 = 868$ $\oplus$'s, or 1604 cycles per 128-bit block.

For AES-256, the number of 8-bit operations is computed as follows: $16 \cdot 14 = 224$ S-box look-ups, $16 \cdot 2 \cdot 13 = 416$ xtime operations, $16 \cdot 15 + 4 \cdot 15 \cdot 13 = 1020$ $\oplus$'s, or 1884 cycles per 128-bit block.

The AES requires only 256 bytes for table lookup because the x^{-1} operation is its own inverse; only the affine operation following the x^{-1} transformation needs to be changed accordingly. For the MESH variants, two S-boxes are needed, one representing discrete exponentiation and another for discrete logarithm in $\mathrm{GF}(2^8 + 1)$.

[6] Only for the key schedule of MESH variants.

Table 1. Parameters and performance figures for 8-bit word-oriented MESH variants and AES

Cipher	MESH-64(8)	MESH-128(8)	AES-128	AES-192	AES-256
Structure	IDEA	IDEA	SPN	SPN	SPN
Block Size (bits)	64	128	128	128	128
Key Size (bits)	128	256	128	192	256
# Rounds	8.5	8.5	10	12	14
Word Size (bits)	8	8	8	8	8
Date	2003	2003	1997	1997	1997
(Encryption) Operators	$\oplus$, $\odot$, $\boxplus$	$\oplus$, $\odot$, $\boxplus$	$\oplus$, S-box, xtime	$\oplus$, S-box, xtime	$\oplus$, S-box, xtime
Subkeys mixed via	$\odot$, $\boxplus$	$\odot$, $\boxplus$	$\oplus$	$\oplus$	$\oplus$
# $\odot$ per block	68	136	—	—	—
# $\boxplus$ per block	68	136	—	—	—
# $\oplus$ per block	96	192	716	868	1020
# S-box lookups	—	—	160	192	224
# xtime per block	—	—	288	352	416
# Subkey Bytes per block	88	144	176	208	240
Diffusion Rate	1 round	1 round	2 rounds	2 rounds	2 rounds
Encryption of 128 bits (# CPU cycles)(†)	600	600	1324	1604	1884
Encryption Key Schedule (# CPU cycles/block)(‡)	376	736	240	288	336
RAM Space for Encryption (bytes)	560	608	304	312	320
1st Subkey depending on all user key bytes	$Z_2^{(3)}$	$Z_3^{(4)}$	$W[20]$	$W[30]$	$W[40]$
Total Encryption Time (†+‡)	976	1336	1564	1892	2220

7 Conclusions

The high flexibility of the MA-box of IDEA was the key factor to allow larger block size variants, such as the 16-bit word-oriented MESH ciphers [22]. This report described new variants of the MESH ciphers that operate on 8-bit words, and that are estimated to be faster than the AES on 8-bit platforms such as the 8051 (smart card). The main design features of the 16-bit word-oriented MESH ciphers were preserved in the 8-bit word-oriented MESH variants, namely:

- flexible block sizes in increments of 16 bits; two variants were described, MESH-64(8) and MESH-128(8), but ciphers with other block sizes can be defined with 80, 96 bits and larger;
- larger MA-boxes that are bijective mappings for any fixed internal subkeys (in order to avoid non-surjective attacks [25]);
- asymmetric key mixing half-rounds, originally designed to avoid slide [6] and advanced slide attacks [5];
- new key schedule algorithms with fast key avalanche that avoid the key overlapping property of IDEA (the main source of weakness exploited in several attacks on the IDEA cipher);
- the same computational graph can be used for both encryption and decryption as in IDEA (just the round subkeys need to be adequately transformed);
- complete diffusion (namely, all output bits depend on all input bits) is achieved in a single round, such as in IDEA;
- absence of explicit S-boxes; therefore the design of the MESH variants does not dependent on strict cryptographic requirements for the S-boxes, such as in the AES and DES [23].
- resistance to Square attacks which are the best known attack on reduced-round versions of the AES [1]. Moreover, preliminary analyses of algebraic attacks indicate that the MESH variants do not possess multivariate quadratic equations, such as found in the AES.

The Square attack complexities on MESH-64(8) and MESH-128(8) are listed in Table 2. Further analyses, including previous attacks on IDEA and the original MESH ciphers, are a work in progress but the same level of security as for the original MESH ciphers is expected, since the main design features of the former were preserved in the latter.

Table 2. Square attack complexities on MESH variants

Cipher	#Rounds	# Chosen Plaintexts	Time
MESH-64(8)	2	2^8	2^{22}
	2.5	2^9	2^{88}
MESH-128(8)	2	2^8	2^{22}
	2.5	$3 \cdot 2^8$	2^{152}
	3	$3 \cdot 2^8$	2^{169}

Acknowledgements

Many thanks to Prof. S.W. Song of the CS Dept. of the Institute of Mathematics and Statistics of the University of São Paulo, Brazil, for the kind logistical support for this research, and to the anonymous referees for the many useful comments.

References

1. AES, The Advanced Encryption Standard Development Process, 1997, http://csrc.nist.gov/encryption/aes/.
2. E. Biham, A. Biryukov, A. Shamir, "Miss-in-the-Middle Attacks on IDEA, Khufu and Khafre," 6th Fast Software Encryption Workshop, L.R. Knudsen, Ed., LNCS 1636, 1999, Springer-Verlag, 124–138.
3. A. Biryukov, C. De Cannière, "Block Ciphers and Systems of Quadratic Equations," 10th Fast Software Encryption Workshop, T. Johansson, Ed., LNCS 2887, 2003, Springer-Verlag, 274–289.
4. A. Biryukov, J. Nakahara,Jr, B. Preneel, J. Vandewalle, "New Weak-Key Classes of IDEA," ICICS 2002, R. Deng and S. Qing and F. Bao and J. Zhou, Eds., LNCS 2513, 2002, Springer-Verlag, 315–326.
5. A. Biryukov, D. Wagner, "Advanced Slide Attacks," Adv. in Cryptology, Eurocrypt'00, B. Preneel,Ed., LNCS 1807, 2000, Springer-Verlag, 589–606.
6. A. Biryukov, D. Wagner, "Slide Attacks," 6th Fast Software Encryption Workshop, L.R. Knudsen,Ed., LNCS 1636, 1999, Springer-Verlag, 245–259.
7. J. Borst, "Differential-Linear Cryptanalysis of IDEA," ESAT Dept., COSIC group, Technical Report 96-2, 1996.
8. CRYPTREC, Evaluation of Cryptographic Techniques Project, 2000-2003, http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html.
9. N.T. Courtois, "Higher Order Correlation Attacks, XL Algorithm and Cryptanalysis of Toyocrypt," Proc. 5th International Conference, ICISC 2002, Seoul, Korea, P.J. Lee and C.H. Lim,Eds., LNCS 2587, 2002, Springer-Verlag, 182–199.
10. N.T. Courtois, J. Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Quadratic Equations," Adv. in Cryptology, Asiacrypt'02, Y. Zheng, Ed., LNCS 2501, 2002, Springer-Verlag, 267–287.
11. J. Daemen, R. Govaerts, J. Vandewalle, "Weak Keys for IDEA," Adv. in Cryptology, Crypto'93, D.R. Stinson, Ed., LNCS 773, 1994, Springer-Verlag, 224–231.
12. J. Daemen, L.R. Knudsen, V. Rijmen, "The Block Cipher SQUARE," 4th Fast Software Encryption Workshop, E. Biham,Ed., LNCS 1267, 1997, Springer-Verlag, 149–165.
13. J. Daemen, V. Rijmen, "AES Proposal: Rijndael," 1st AES Conference, California, USA, 1998, http://www.nist.gov/aes
14. H. Demirci, "Square-like Attacks on Reduced Rounds of IDEA," 9th Selected Areas in Cryptography Workshop, SAC'02, K. Nyberg and H. Heys, Eds., LNCS 2595, 2002, Springer-Verlag, 147–159.
15. H. Demirci, E. Ture, A.A. Selçuk, "A New Meet-in-the-Middle Attack on the IDEA block cipher," 10th Selected Areas in Cryptography Workshop, SAC'03, M. Matsui and R. Zuccherato, Eds., LNCS 3006, 2003, Springer-Verlag.
16. S. Garfinkel, "PGP: Pretty Good Privacy," O'Reilly and Associates, 1994.

17. P.M. Hawkes, "Asymptotic Bounds on Differential Probabilities and an Analysis of the Block Cipher IDEA," The University of Queensland, St. Lucia, Australia, Dec, 1998.
18. X. Lai, "On the Design and Security of Block Ciphers," ETH Series in Information Processing, J.L. Massey, Ed., vol. 1, 1995, Hartung-Gorre Verlag, Konstanz.
19. X. Lai, J.L. Massey, S. Murphy, "Markov Ciphers and Differential Cryptanalysis," Adv. in Cryptology, Eurocrypt'91, D.W. Davies, Ed., LNCS 547, 1991, Springer-Verlag, 17–38.
20. W. Meier, "On the Security of the IDEA Block Cipher," Adv. in Cryptology, Eurocrypt'93, T. Helleseth, Ed., LNCS 765, 1994, Springer-Verlag, 371–385.
21. A.J. Menezes, P.C. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography," CRC Press.
22. J. Nakahara,Jr, V. Rijmen, B. Preneel, J. Vandewalle, "The MESH Block Ciphers," The 4th International Workshop on Info. Security Applications, WISA 2003, K. Chae and M. Yung,Eds., LNCS 2908, 2003, Springer-Verlag, 458–473.
23. NBS, Data Encryption Standard (DES)," FIPS PUB 46, Federal Information Processing Standards Publication 46, U.S. Department of Commerce, Jan, 1977.
24. NESSIE, New European Schemes for Signatures, Integrity and Encryption, 2000, http://cryptonessie.org.
25. V. Rijmen, B. Preneel, E. De Win, "On Weaknesses of Non-Surjective Round Functions," Design, Codes and Cryptography, vol. 12, number 3, 1997, 253–266.
26. H.M. Yıldırım, "Some Linear Relations for Block Cipher IDEA," The Middle East Technical University, Jan, 2002.

Related-Key Attacks on Reduced Rounds of SHACAL-2*

Jongsung Kim[1], Guil Kim[1], Sangjin Lee[1],
Jongin Lim[1], and Junghwan Song[2]

[1] Center for Information Security Technologies(CIST),
Korea University, Anam Dong, Sungbuk Gu,
Seoul, Korea
{joshep, okim912, sangjin, jilim}@cist.korea.ac.kr
[2] Department of Mathematics, Hanyang University,
17 Haengdangdong Seongdongku, Seoul
camp123@hanyang.ac.kr

Abstract. SHACAL-2 is a 256-bit block cipher with up to 512 bits of key length based on the hash function SHA-2. It was submitted to the NESSIE project and was recommended as one of the NESSIE selections. In this paper we present two types of related-key attacks called the related-key differential-(non)linear and the related-key rectangle attacks, and we discuss the security of SHACAL-2 against these two types of attacks. Using the related-key differential-nonlinear attack we can break SHACAL-2 with 512-bit keys up to 35 out of its 64 rounds, and using the related-key rectangle attack we can break SHACAL-2 with 512-bit keys up to 37 rounds.

1 Introduction

SHACAL-2 [5] is a 256-bit block cipher based on the compression function of the hash function SHA-2. The cipher has 64 rounds and supports a variable key length up to 512 bits. Recently, the SHACAL-2 cipher was recommended as one of the NESSIE (New European Schemes for Signatures, Integrity, and Encryption) selections.

The best cryptanalytic result obtained on SHACAL-2 so far is the analysis of a differential-nonlinear attack on 32-round SHACAL-2 [15]. The attack presented in [15] uses a nontrivial 3-round nonlinear relation of SHACAL-2. This nonlinear relation is also used to mount our attack on SHACAL-2.

In this paper we present two types of related-key attacks called the related-key differential-(non)linear and the related-key rectangle attacks, and we discuss the security of SHACAL-2 against these two types of attacks.

* This work was supported by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 175–190, 2004.

Table 1. Comparison of our results with the previous attacks on SHACAL-2

Type of Attack	Number of Rounds	Complexity Data / Time / Memory
Impossible Differential	30	744CP / $2^{495.1}$ / $2^{14.5}$ [8]
Differential-Nonlinear	32	$2^{43.4}$CP / $2^{504.2}$ / $2^{48.4}$ [15]
Square-Nonlinear	28	$463 \cdot 2^{32}$CP / $2^{494.1}$ / $2^{45.9}$ [15]
Related-Key Differential-Nonlinear	35	$2^{42.32}$RK-CP / $2^{452.10}$ / $2^{47.32}$ (New)
Related-Key Rectangle	37	$2^{233.16}$RK-CP / $2^{484.95}$ / $2^{238.16}$ (New)

CP: Chosen Plaintexts, RK-CP: Related-Key Chosen Plaintexts,
Time: Encryption units, Memory : Bytes of memory

The related-key differential-linear attack was introduced in [7]. The attack presented in [7] uses a related-key differential-linear distinguisher with a probability of 1. In this paper, however, we enhance this technique into the cases where the probability of distinguisher is smaller than 1, and we extend it into a technique called the related-key differential-nonlinear attack which uses a related-key differential-nonlinear distinguisher. Using the related-key differential-nonlinear distinguisher we can attack 35-round SHACAL-2 with a data complexity of $2^{42.32}$ related-key chosen plaintexts and a time complexity of $2^{452.1}$ encryptions, which is faster than exhaustive key search.

The related-key rectangle attack was introduced in [10]. According to [10], two types of distinguishers can be used in this attack. In this paper we present one of the two types of distinguishers, which is efficiently used to analyze the SHACAL-2 cipher. Using the related-key rectangle distinguisher we can attack 37-round SHACAL-2 with a data complexity of $2^{233.16}$ related-key chosen plaintexts and a time complexity of $2^{484.95}$ encryptions, which is faster than exhaustive key search. See Table 1 for a summary of our results and their comparison with the previous attacks.

This paper is organized as follows: In Section 2 we describe the block cipher SHACAL-2. In Section 3 we present the related-key differential-(non)linear and the related-key rectangle attacks, which can be useful tools to analyze block ciphers. Section 4 presents our related-key differential-nonlinear attack on 35-round SHACAL-2 and Section 5 presents our related-key rectangle attack on 37-round SHACAL-2. Finally, Section 6 concludes the paper.

2 Preliminaries

In this Section we present some notations which are used throughout this paper, and we briefly describe the SHACAL-2 cipher, and we summarize a 3-round nonlinear relation of SHACAL-2 presented in [8, 15], which is used in our related-key differential-nonlinear attack on SHACAL-2.

2.1 Notations

We use the following notations, where the right most bit is referred to as the 0-th bit, i.e., the least significant bit.

- P^r : A 256-bit input data of the r^{th} round, i.e., $P^r = (A^r, B^r, \cdots, H^r)$.
- P : A 256-bit plaintext, i.e., $P = (A, B, \cdots, H) = (A^0, B^0, \cdots, H^0) = P^0$.
- x_i^r : The i^{th} bit of 32-bit word X^r where $X^r \in \{A^r, B^r, \cdots, H^r, W^r, K^r, T_1^r\}$ (The notations W^r, K^r, T_1^r will be defined in the next subsection.)
- ? : A 32-bit unknown value.
- e_i : A 32-bit word that has $0's$ in all bit positions except for bit i.
- $e_{i_1,\cdots,i_k}$: $e_{i_1} \oplus \cdots \oplus e_{i_k}$.
- $e_{i_1,\cdots,i_k,\sim}$: A 32-bit word that has $1's$ in the positions of bits $i_1, \cdots, i_k$, and unconcerned values in the positions of bits $(i_k + 1) \sim 31$, and $0's$ in all other bit positions, where $i_1 < \cdots < i_k$. (The unconcerned value can be 0, 1 or an unknown value.)

2.2 Description of SHACAL-2

SHACAL-2 [5] is a 256-bit block cipher which is based on the compression function of the hash function SHA-2 [17]. It is composed of 64 rounds and supports a variable key length up to 512 bits. However, SHACAL-2 is not intended to be used with a key shorter than 128 bits.

According to our notations, a 256-bit plaintext P is divided into $A^0, B^0, C^0, D^0, E^0, F^0, G^0$ and H^0, and the corresponding ciphertext C is composed of $A^{64}, B^{64}, C^{64}, D^{64}, E^{64}, F^{64}, G^{64}$ and H^{64}. The r^{th} round of encryption is performed as follows.

$$
\begin{aligned}
T_1^{r+1} &= H^r + \Sigma_1(E^r) + Ch(E^r, F^r, G^r) + K^r + W^r \\
T_2^{r+1} &= \Sigma_0(A^r) + Maj(A^r, B^r, C^r) \\
H^{r+1} &= G^r \\
G^{r+1} &= F^r \\
F^{r+1} &= E^r \\
E^{r+1} &= D^r + T_1^{r+1} \\
D^{r+1} &= C^r \\
C^{r+1} &= B^r \\
B^{r+1} &= A^r \\
A^{r+1} &= T_1^{r+1} + T_2^{r+1}
\end{aligned}
$$

for $r = 0, ..., 63$ where $+$ means the addition modulo 2^{32} of 32-bit words, W^r are the 32-bit round subkeys, and K^r are the 32-bit round constants which are different in each of the 64 rounds. The functions used in the above encryption process are defined as follows.

$$
\begin{aligned}
Ch(X, Y, Z) &= (X\&Y) \oplus (\neg X\&Z) \\
Maj(X, Y, Z) &= (X\&Y) \oplus (X\&Z) \oplus (Y\&Z) \\
\Sigma_0(X) &= S_2(X) \oplus S_{13}(X) \oplus S_{22}(X) \\
\Sigma_1(X) &= S_6(X) \oplus S_{11}(X) \oplus S_{25}(X)
\end{aligned}
$$

where $\neg X$ means the complement of 32-bit word X and $S_i(X)$ means the right rotation of X by i bit positions.

The key scheduling algorithm of SHACAL-2 is performed based on a linear shift feedback register. It accepts a maximum 512-bit key and shorter keys than 512 bits are used by padding the key with zeros to a 512-bit string. Let the 512-bit key string be denoted $W = [W^0||W^1||\cdots||W^{15}]$. The key expansion of 512 bits W to 2048 bits is defined by

$$W^r = \sigma_1(W^{r-2}) + W^{r-7} + \sigma_0(W^{r-15}) + W^{r-16},\ 16 \le r \le 63.$$
$$\sigma_0(x) = S_7(x) \oplus S_{18}(x) \oplus R_3(x)$$
$$\sigma_1(x) = S_{17}(x) \oplus S_{19}(x) \oplus R_{10}(x)$$

where $R_i(X)$ means the right shift of 32-bit word X by i bit positions.

2.3 A 3-Round Nonlinear Relation of SHACAL-2

This subsection summarizes the 3-round nonlinear relation of SHACAL-2 presented in [8, 15].

The value h_0^r can be represented as the output of nonlinear function $NF(A^{r+3}, B^{r+3}, \cdots, H^{r+3}, K^r, K^{r+1}, K^{r+2}, W^r, W^{r+1}, W^{r+2})$, denoted NF^{r+3}, where $0 \le r \le 61$.

$$\begin{aligned} h_0^r = {} & c_0^{r+3} \oplus d_2^{r+3} \oplus d_{13}^{r+3} \oplus d_{22}^{r+3} \oplus (d_0^{r+3} \& (e_0^{r+3} \oplus t_{1,0}^{r+3})) \oplus (d_0^{r+3} \& (f_0^{r+3} \oplus t_{1,0}^{r+2})) \\ & \oplus ((e_0^{r+3} \oplus t_{1,0}^{r+3}) \& (f_0^{r+3} \oplus t_{1,0}^{r+2})) \oplus h_6^{r+3} \oplus h_{11}^{r+3} \oplus h_{25}^{r+3} \\ & \oplus (h_0^{r+3} \& h_0^{r+2}) \oplus ((\neg h_0^{r+3}) \& h_0^{r+1}) \oplus k_0^r \oplus w_0^r \end{aligned}$$

The values h_0^{r+1}, $t_{1,0}^{r+2}$, h_0^{r+2} and $t_{1,0}^{r+3}$ in the above equation are represented as follows.

$$\begin{cases} t_{1,0}^{r+3} = a_0^{r+3} \oplus b_2^{r+3} \oplus b_{13}^{r+3} \oplus b_{22}^{r+3} \oplus (b_0^{r+3} \& c_0^{r+3}) \oplus (b_0^{r+3} \& d_0^{r+3}) \oplus (c_0^{r+3} \& d_0^{r+3}) \\ t_{1,0}^{r+2} = b_0^{r+3} \oplus c_2^{r+3} \oplus c_{13}^{r+3} \oplus c_{22}^{r+3} \oplus (c_0^{r+3} \& d_0^{r+3}) \oplus (c_0^{r+3} \& (e_0^{r+3} \oplus t_{1,0}^{r+3})) \oplus \\ \qquad (d_0^{r+3} \& (e_0^{r+3} \oplus t_{1,0}^{r+3})) \\ h_0^{r+2} = t_{1,0}^{r+3} \oplus f_6^{r+3} \oplus f_{11}^{r+3} \oplus f_{25}^{r+3} \oplus (f_0^{r+3} \& g_0^{r+3}) \oplus ((\neg f_0^{r+3}) \& h_0^{r+3}) \oplus k_0^{r+2} \oplus w_0^{r+2} \\ h_0^{r+1} = t_{1,0}^{r+2} \oplus g_6^{r+3} \oplus g_{11}^{r+3} \oplus g_{25}^{r+3} \oplus (g_0^{r+3} \& h_0^{r+3}) \oplus ((\neg g_0^{r+3}) \& h_0^{r+2}) \oplus k_0^{r+1} \oplus w_0^{r+1} \end{cases}$$

In order to facilitate the description of our related-key differential-nonlinear attack on SHACAL-2, we use a modified notation MNF^{r+3} such that $MNF^{r+3} = NF^{r+3} \oplus k_0^r \oplus w_0^r$.

3 Related-Key Attacks

This Section presents two types of related-key attacks which can be useful tools to analyze block ciphers. Firstly, we describe the related-key differential-linear

attack and extend it into a technique called the related-key differential-nonlinear attack. Secondly, we describe the related-key rectangle attack.

3.1 The Related-Key Differential-(Non)Linear Attack

In 1994, Langford and Hellman [12] showed that the differential and the linear attacks can be combined together by a technique called the differential-linear attack. This attack uses a differential with probability 1 and a linear approximation with bias q. In [3], Biham, Dunkelman and Keller presented an enhanced differential-linear attack which can be applied to the cases where the probability of differential part is smaller than 1. More generally, in [15], Shin et al. showed that the differential-linear attack can be extended to the cases where a square characteristic is used instead of a differential when the value of q is very close to $1/2$ or equal to $1/2$. Furthermore, these attacks can be extended to the cases where a nonlinear approximation is used instead of a linear approximation [15].

In 1998, Hawkes [7] presented the related-key differential-linear attack which is a combination of the related-key and the differential-linear attacks. The attack presented in [7] uses a related-key differential with probability 1 and a linear approximation with bias $\frac{1}{2}$. However, we can enhance this technique into the general cases where the probability of related-key differential is less than or equal to 1 and the bias of linear approximation is less than or equal to $\frac{1}{2}$. Furthermore, we can extend it into a technique called the related-key differential-nonlinear attack which uses a related-key differential-nonlinear distinguisher.

The related-key differential-linear attack requires the encryptions of plaintext pairs P and P^* under keys k and k^*, respectively, where k and k^* be different, but related keys. We use the notations Ω_P, Ω_T to present the input and output differences of a differential, and λ_P, λ_T, λ_K to present the input, output and subkey bit masks of a linear approximation, respectively. Let a block cipher $E_k : \{0,1\}^n \rightarrow \{0,1\}^n$ be composed of a cascade $E_k = E_k^1 \circ E_k^0$ where k is a master key of the cipher. A block cipher E_k can be also denoted $E = E^1 \circ E^0 : \{0,1\}^{|k|} \times \{0,1\}^n \rightarrow \{0,1\}^n$. E^0 is a subcipher for the related-key differential $\Omega_P \rightarrow \Omega_T$ with probability $p^* \leq 1$ (i.e., $Pr_X[E_k^0(X) \oplus E_{k^*}^0(X^*) = \Omega_T | X \oplus X^* = \Omega_P] = p^*$) and E^1 is a subcipher for the linear approximation $\lambda_P \rightarrow \lambda_T$ with probability $\frac{1}{2} + q$ (i.e., $Pr_X[\lambda_P \cdot X \oplus \lambda_T \cdot E^1(X) \oplus \lambda_K \cdot K = 0] = \frac{1}{2} + q$ where K be the subkey of the E_1 subcipher).

In case the plaintext pair P and P^* satisfies the related-key differential $\Omega_P \rightarrow \Omega_T$ (related to probability p^*), we get the one bit equation $\lambda_P \cdot (E_k^0(P) \oplus E_{k^*}^0(P^*)) = a$ with probability 1 where $a = \lambda_P \cdot \Omega_T$. In case the plaintext pair P and P^* does not satisfy the related-key differential (related to probability $1 - p^*$), we assume that $\lambda_P \cdot (E_k^0(P) \oplus E_{k^*}^0(P^*))$ follows a random behavior. Thus, in the above two cases, we get the one bit equation

$$\lambda_P \cdot (E_k^0(P) \oplus E_{k^*}^0(P^*)) = a \tag{1}$$

with probability $\frac{1}{2} + \frac{p^*}{2} (= p^* \cdot 1 + (1 - p^*) \cdot \frac{1}{2})$. According to our assumption, we have also the following two linear approximations

$$\lambda_P \cdot E_k^0(P) \oplus \lambda_T \cdot E_k^1(E_k^0(P)) \oplus \lambda_K \cdot K = 0 \tag{2}$$

$$\lambda_P \cdot E^0_{k^*}(P^*) \oplus \lambda_T \cdot E^1_{k^*}(E^0_{k^*}(P^*)) \oplus \lambda_K \cdot K^* = 0 \tag{3}$$

with probability $\frac{1}{2} + q$, respectively. Hence, applying (1), (2), (3) to the basic linear cryptanalytic method presented in [13] (i.e, summing over (1), (2), (3)), we have the following equation

$$\lambda_T \cdot E^1_k(E^0_k(P)) \oplus \lambda_T \cdot E^1_{k^*}(E^0_{k^*}(P^*)) \oplus \lambda_K \cdot K \oplus \lambda_K \cdot K^* = a \tag{4}$$

with probability $\frac{1}{2} + 2p^*q^2 (= \frac{1}{2} + 2^{3-1} \cdot \frac{p^*}{2} \cdot q^2)$. That is, we have the following equation

$$\lambda_T \cdot E_k(P) \oplus \lambda_T \cdot E_{k^*}(P^*) = 0 \tag{5}$$

with bias $2p^*q^2$. So the attack using the above related-key differential-linear distinguisher requires $O(p^{*-2}q^{-4})$ related-key chosen plaintexts to succeed.

As stated above, this attack can be extended into the cases where a nonlinear approximation is used instead of a linear approximation. We call such a distinguisher a *related-key differential-nonlinear distinguisher*. In this paper we exploit a related-key differential-nonlinear distinguisher to mount our attack on SHACAL-2.

3.2 The Related-Key Rectangle Attack

In 2004, Kim et al. [10] presented the related-key rectangle attack which is a combination of the related-key and the rectangle attacks. The main idea of the related-key rectangle attack is to use consecutive two differentials composed of which the first one is a related-key differential and the second one is a differential. Therefore, this attack is very useful when we have a good related-key differential followed by a good differential.

Let $E_k = E^1_k \circ E^0_k$ (or $E = E^1 \circ E^0$) be a block cipher as described above. We assume that for E^0 we have a related-key differential $\alpha \rightarrow \beta$ with probability p^*_β (i.e., $Pr_X[E^0_k(X) \oplus E^0_{k^*}(X^*) = \beta | X \oplus X^* = \alpha] = p^*_\beta$ where k and k^* be different, but related keys), and for E^1 we have a differential $\gamma \rightarrow \delta$ with probability q_γ (i.e., $Pr_X[E^1(X) \oplus E^1(X') = \delta | X \oplus X' = \gamma] = q_\gamma$).

The related-key rectangle distinguisher is based on building quartets of plaintexts (P_i, P^*_i, P_j, P^*_j) which satisfy several differential conditions. Assume that P_i, P_j are encrypted under E_k and P^*_i, P^*_j are encrypted under E_{k^*} such that $P_i \oplus P^*_i = P_j \oplus P^*_j = \alpha$. We denote by X_i, X^*_i, X_j, X^*_j the encrypted values of P_i, P^*_i, P_j, P^*_j under E^0, respectively, and by C_i, C^*_i, C_j, C^*_j the encrypted values of X_i, X^*_i, X_j, X^*_j under E^1, respectively. We are interested in the cases where $X_i \oplus X^*_i = X_j \oplus X^*_j = \beta$ and $X_i \oplus X_j = \gamma$ as in these cases $X^*_i \oplus X^*_j = (X_i \oplus \beta) \oplus (X_j \oplus \beta) = \gamma$ as well. If both the C_i, C_j pair and the C^*_i, C^*_j pair satisfy a δ difference, a quartet satisfying all these differential conditions is called a right quartet. A description of such a quartet is shown in Fig. 1. More generally, a right quartet represents one which satisfies any β and γ difference conditions for given α and δ differences.

If we have m pairs with difference α where one plaintext of each pair is encrypted under E_k and the other plaintext is encrypted under E_{k^*}, then we

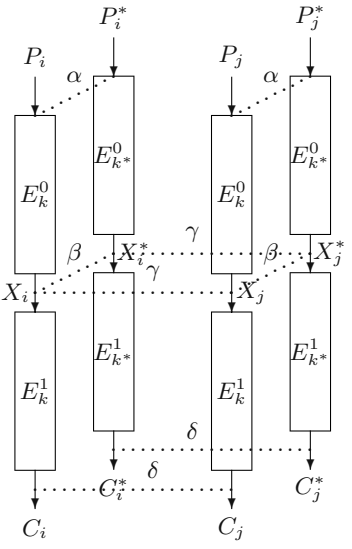

Fig. 1. A Related-Key Rectangle Distinguisher

have about mp_β^* pairs satisfying the related-key differential $\alpha \to \beta$ for E^0. The mp_β^* pairs generate about $\frac{(mp_\beta^*)^2}{2}$ quartets consisting of two such pairs. Assuming that the intermediate encryption values distribute uniformly over all possible values, we get $X_i \oplus X_j = \gamma$ with a probability of 2^{-n}. As stated above, once this occurs we get $X_i^* \oplus X_j^* = \gamma$ with a probability of 1. Since each of the pairs (X_i, X_j) and (X_i^*, X_j^*) satisfies the differential $\gamma \to \delta$ for E^1 with probability q_γ, the expected number of right quartets is

$$\sum_{\beta,\gamma} \frac{(mp_\beta^*)^2}{2} \cdot 2^{-n} \cdot (q_\gamma)^2 = m^2 \cdot 2^{-n-1} \cdot (\hat{p^*})^2 \cdot (\hat{q})^2$$

where $\hat{p^*} = (\sum_\beta (p_\beta^*)^2)^{\frac{1}{2}}$, $\hat{q} = (\sum_\gamma (q_\gamma)^2)^{\frac{1}{2}}$.

On the other hand, for a random permutation the expected number of right quartets is $m^2 \cdot 2^{-2n-1}$, as $p_\beta^* = 2^{-n}$ and $q_\gamma = 2^{-n}$. Therefore, if $\hat{p^*} \cdot \hat{q} > 2^{-n/2}$ and m is sufficiently large, we can distinguish between E and a random permutation.

4 Related-Key Differential-Nonlinear Attack on 35-Round SHACAL-2

In this Section we describe a 28-round related-key differential-nonlinear distinguisher of SHACAL-2 and design our related-key differential-nonlinear attack on 35-round SHACAL-2.

As stated in subsection 2.2, the key scheduling algorithm of SHACAL-2 is performed based on a linear feedback shift register. However, this key schedul-

ing algorithm has low difference propagations for the first several round keys. That is, in case the related keys are same each other except for the sixth round key W^6, the expanded round keys $W^{16}, W^{17}, \cdots, W^{20}$ have all zero differences and W^{21}, W^{22} have the $e_{13,\sim}$ and the e_{31} differences, respectively. These difference propagations of related keys enable us to make a 25-round related-key truncated differential with a high probability. Namely, we can construct a 25-round related-key truncated differential $\Omega_P \rightarrow \Omega_T$ for rounds $0 \sim 24$ (E^0) with probability 2^{-16}, where $\Omega_P = (0, e_{31}, 0, 0, e_{6,20,25}, 0, 0, e_{9,13,19})$ and $\Omega_T = (?, ?, ?, e_{13,\sim}, ?, ?, ?, e_{13,\sim})$. See Tables 2, 3 for the details of this differential. Note that this related-key truncated differential requires plaintext pairs (P, P^*) with 8-bit fixed values as depicted in Table 3.

Table 2. A Related-Key Truncated Differential for E^0

Round (i)	ΔA^i	ΔB^i	ΔC^i	ΔD^i	ΔE^i	ΔF^i	ΔG^i	ΔH^i	ΔW^i	Prob.
0	0	e_{31}	0	0	$e_{6,20,25}$	0	0	$e_{9,13,19}$	0	2^{-3}
1	0	0	e_{31}	0	0	$e_{6,20,25}$	0	0	0	2^{-4}
2	0	0	0	e_{31}	0	0	$e_{6,20,25}$	0	0	2^{-3}
3	0	0	0	0	e_{31}	0	0	$e_{6,20,25}$	0	2^{-4}
4	0	0	0	0	0	e_{31}	0	0	0	2^{-1}
5	0	0	0	0	0	0	e_{31}	0	0	2^{-1}
6	0	0	0	0	0	0	0	e_{31}	e_{31}	1
7	0	0	0	0	0	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
20	0	0	0	0	0	0	0	0	0	1
21	0	0	0	0	0	0	0	0	$e_{13,\sim}$	1
22	$e_{13,\sim}$	0	0	0	$e_{13,\sim}$	0	0	0	e_{31}	1
23	?	$e_{13,\sim}$	0	0	?	$e_{13,\sim}$	0	0	?	1
24	?	?	$e_{13,\sim}$	0	?	?	$e_{13,\sim}$	0	?	1
25	?	?	?	$e_{13,\sim}$	?	?	?	$e_{13,\sim}$		

Table 3. Some fixed bits of the P, P^* pair

A, A^*	C, C^*	F, F^*	G, G^*
$a_{31} = a_{31}^* = 0$	$c_{31} = c_{31}^* = 0$	$f_6 = f_6^* = 0$	$g_6 = g_6^* = 0$
		$f_{20} = f_{20}^* = 0$	$g_{20} = g_{20}^* = 0$
		$f_{25} = f_{25}^* = 0$	$g_{25} = g_{25}^* = 0$

We use our 25-round related-key truncated differential to build a distinguisher with a probability of $\frac{1}{2} + 2^{-17} (= 2^{-16} + \frac{1}{2} \cdot (1 - 2^{-16}))$. The distinguisher exploits partial information of the differential, i.e., the distinguisher exploits the fact that if the plaintext pairs P, P^* have the $(0, e_{31}, 0, 0, e_{6,20,25}, 0, 0, e_{9,13,19})$ difference and the foregoing 8-bit fixed values, then it holds that $h_0^{25} = h_0^{*25}$ with a probability of $\frac{1}{2} + 2^{-17}$. This approximation assumes that the behavior

of the remaining fraction of $1-2^{-16}$ of the pairs follows uniform distribution. In order to verify the probability $\frac{1}{2}+2^{-17}$ we performed a series of 5 simulations using 2^{34} plaintext pairs each (for any two simulations we used different random related keys and different plaintext pairs). Since the 25-round distinguisher has a probability of $\frac{1}{2}+2^{-17}$, we expect about $2^{33}+131072(=2^{34}\cdot(\frac{1}{2}+2^{-17}))$ plaintext pairs which satisfy $h_0^{25}=h_0^{*25}$ in 2^{34} plaintext pairs. Our simulation results support our expectation, i.e, we obtained such plaintext pairs of $2^{33}+128629, 2^{33}+130921, 2^{33}+138897, 2^{33}+143916, 2^{33}+145975$ from our simulations. Based on these simulations we are convinced that the probability of the 25-round distinguisher is approximately $\frac{1}{2}+2^{-17}$.

In order to get a stronger distinguisher we concatenate the forgoing 3-round nonlinear relation to the above distinguisher. Since given the P, P^* pairs it holds that $h_0^{25}=h_0^{*25}$ with a probability of approximately $\frac{1}{2}+2^{-17}$, we have the equation $NF^{28}=NF^{*28}$ with a probability of approximately $\frac{1}{2}+2^{-17}$. Equivalently, we have the following equation

$$MNF^{28}=MNF^{*28} \tag{6}$$

with a bias of approximately 2^{-17}. Thus, we have a 28-round related-key differential-nonlinear distinguisher with a bias of approximately 2^{-17}.

We now present a method to use the 28-round related-key differential-nonlinear distinguisher to find a master key pair of 35-round SHACAL-2 where the cipher uses related keys whose difference is $(0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0,\ 0,0)$. The attack procedure is performed as follows.

1. Prepare 5 pools of 2^{39} plaintext pairs $(P_{i,j},P_{i,j}^*)$, $i=0,1,\cdots,4$, $j=0,1,\cdots,2^{39}-1$, that have the Ω_P difference and the 8-bit fixed values. We choose all different plaintext pairs for any two pools $i's$. (Note that each $P_{i,j}$ is encrypted using a key k, and each $P_{i,j}^*$ is encrypted using a key k^* where k and k^* have a difference $(0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0,0,0)$.) Encrypt all these plaintext pairs to get the 5 pools of 2^{39} ciphertext pairs $(C_{i,j},C_{i,j}^*)$.
2. Guess a 207-bit subkey pair (sk,sk^*). A subkey sk represents W^{34},W^{33},W^{32}, $W^{31},w_0^{30},w_1^{30},\cdots,w_{25}^{30},w_0^{29},\ w_1^{29},\cdots,\ w_{25}^{29}\ ,w_0^{28},w_1^{28},\cdots,w_{24}^{28},w_0^{27},\ w_0^{26}$ and the other subkey sk^* represents W^{*34}, W^{*33},W^{*32}, W^{*31},w_0^{*30},w_1^{*30},$\cdots$,w_{25}^{*30}, w_0^{*29},w_1^{*29},$\cdots$,w_{25}^{*29},w_0^{*28},$w_1^{*28}\ ,\cdots,\ w_{24}^{*28},\ w_0^{*27}$,$w_0^{*26}$. (Note that it is sufficient to guess this 207-bit subkey pair for computing the value ΔMNF^{28} from a given related-key ciphertext pair of 35-round SHACAL-2.)
3. For $i=0$ to 4 do the following :
   (a) Partially decrypt all 2^{39} ciphertexts $C_{i,j}$ (resp., $C_{i,j}^*$) using the sk subkey (resp., the sk^* subkey), and check Equation (6). If the number of ciphertext pairs satisfying Equation (6) is greater than $2^{38}-2^{21.6}$ and less than $2^{38}+2^{21.6}$, then go to Step 2.
4. For the suggested subkey sk, do an exhaustive search for the 305-bit remaining keys using trial encryption (For the suggested subkey sk, we use two known plaintext and ciphertext pairs for the trial encryption). If a 512-bit key k' is suggested, output the k' key as a master key of 35-round SHACAL-2. In

this case, we also output the key $k' \oplus (0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0,0,0)$ as the related master key of 35-round SHACAL-2. Otherwise, go to Step 2. In case 207-bit subkey pairs (sk, sk^*) are all tested and there does not exist a suggested key k', we stop this algorithm without output.

The data complexity of this attack is about $2^{42.32}(\approx 5 \cdot 2 \cdot 2^{39})$ related-key chosen plaintexts. The memory requirements of this attack are dominated by the memory for ciphertext pairs $(C_{i,j}, C^*_{i,j})$ and thus this attack requires about $2^{47.32}(5 \cdot 2 \cdot 2^{39} \cdot \frac{256}{8})$ memory bytes.

We now analyze the time complexity of this attack. The time complexity of Step 1 (the data collecting step) is about $2^{42.32}(\approx 5 \cdot 2 \cdot 2^{39})$ 35-round SHACAL-2 encryptions. To compute the time complexity of Step 3 we should estimate the survived fraction of subkey pairs (sk, sk^*) with respect to each loop i. For a random permutation each pair behaves randomly, and thus on the average half of the ciphertext pairs satisfy Equation (6) for a wrong subkey pairs. Hence, the number of plaintext pairs satisfying Equation (6) behaves like a binomial random variable $X \sim Bin(2^{39}, \frac{1}{2})$. It is easy to see that a binomial random variable can be approximated according to the normal distribution, and thus $X \sim N(\mu, \sigma^2)$ where $\mu = 2^{38}$ and $\sigma^2 = 2^{37}$, equivalently $Z(= \frac{X-\mu}{\sigma}) \sim N(0,1)$. Since $Pr[X \geq 2^{38} + 2^{21.6}$ or $X \leq 2^{38} - 2^{21.6}] = Pr[Z \geq 8.5742$ or $Z \leq -8.5742] \approx 2^{-53.27}$, the survived fraction of subkey pairs with respect to each loop i is about $2^{-53.27}$. It follows that after the i^{th} loop the number of survived subkey pairs is about $(2^{207})^2 \cdot 2^{-53.27*(i+1)}$. So the time complexity of Step 3 is about $2^{451.64}(\approx \sum_{i=0}^{4} 2^{39} \cdot 2 \cdot (2^{207})^2 \cdot 2^{-53.27*i} \cdot \frac{7}{35})$ 35-round SHACAL-2 encryptions. Since the number of survived subkey pairs in Step 3 is about $2^{147.65}(\approx (2^{207})^2 \cdot 2^{-53.27*5})$, the time complexity of Step 4 is about $2^{450.21}(\approx 2^{147.65} \cdot 2^{305} \cdot \frac{7}{35})$ 35-round SHACAL-2 encryptions. Thus the total time complexity of this attack is about $2^{452.1}(\approx 2^{42.32} + 2^{451.64} + 2^{450.21})$ 35-round SHACAL-2 encryptions.

In order to compute the success rate of this attack we check the probability that the right subkey pair survives in Step 3. For the right subkey pair Equation (6) holds with a probability of approximately $\frac{1}{2} + 2^{-17}$ or $\frac{1}{2} - 2^{-17}$. In case the probability is approximately $\frac{1}{2} + 2^{-17}$, we have $X \sim Bin(2^{39}, \frac{1}{2} + 2^{-17})$ (i.e., $X \sim N(\mu, \sigma^2)$ where $\mu = 2^{38} + 2^{22}$ and $\sigma^2 = \mu \cdot (\frac{1}{2} - 2^{-17})$) where X is the number of plaintext pairs satisfying Equation (6) for the right subkey pair. Using the above analysis we can check the probability that the right subkey pair survives in each loop of Step 3 is about $1 - 2^{-8.34}(\approx Pr[X \geq 2^{38} + 2^{21.6}] = Pr[Z \geq -2.7395])$. It follows that the probability that the right subkey pair survives in Step 3 is about $0.98(\approx (1 - 2^{-8.34})^5)$. In case Equation (6) holds with a probability of approximately $\frac{1}{2} - 2^{-17}$, we have the same result. Therefore, the success rate of this attack is about 98%. Note that our attack algorithm can be converted into the key ranking algorithm presented in [14] with the same complexity and success rate. However, the key ranking algorithm requires a number of memory bytes for all possible 2^{414} subkey pairs (sk, sk^*).

5 Related-Key Rectangle Attack on 37-Round SHACAL-2

This Section describes a 33-round related-key rectangle distinguisher of SHACAL-2 and designs our related-key rectangle attack on 37-round SHACAL-2.

As stated earlier, the key scheduling algorithm of SHACAL-2 has low difference propagations for the first several round keys. In case the related keys are same each other except for the eighth round key W^8, the expanded round keys $W^{16}, W^{17}, \cdots, W^{22}$ have all zero differences. This fact allows us to make a 23-round related-key differential characteristic with a high probability. Namely, we can construct a 23-round related-key differential characteristic $\alpha \rightarrow \beta$ for rounds $0 \sim 22$ (E^0) with probability 2^{-31}, where $\alpha = (0, 0, e_{6,9,18,20,25,29}, e_{31}, 0, e_{9,13,19}, e_{18,29}, e_{31})$ and $\beta = (0, 0, 0, 0, 0, 0, 0, 0)$. See Tables 4, 5 for the details of this characteristic. This related-key differential characteristic requires plaintext pairs (P, P^*) with 22-bit fixed values as depicted in Table 5.

Table 4. A Related-Key Differential Characteristic for E^0 ($M = \{6, 9, 18, 20, 25, 29\}$)

Round (i)	ΔA^i	ΔB^i	ΔC^i	ΔD^i	ΔE^i	ΔF^i	ΔG^i	ΔH^i	ΔW^i	Prob.
0	0	0	e_M	e_{31}	0	$e_{9,13,19}$	$e_{18,29}$	e_{31}	0	1
1	e_{31}	0	0	e_M	0	0	$e_{9,13,19}$	$e_{18,29}$	0	2^{-11}
2	0	e_{31}	0	0	$e_{6,20,25}$	0	0	$e_{9,13,19}$	0	2^{-7}
3	0	0	e_{31}	0	0	$e_{6,20,25}$	0	0	0	2^{-4}
4	0	0	0	e_{31}	0	0	$e_{6,20,25}$	0	0	2^{-3}
5	0	0	0	0	e_{31}	0	0	$e_{6,20,25}$	0	2^{-4}
6	0	0	0	0	0	e_{31}	0	0	0	2^{-1}
7	0	0	0	0	0	0	e_{31}	0	0	2^{-1}
8	0	0	0	0	0	0	0	e_{31}	e_{31}	1
9	0	0	0	0	0	0	0	0	0	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
22	0	0	0	0	0	0	0	0	0	1
23	0	0	0	0	0	0	0	0	·	·

Table 5. Some fixed bits of the P, P^* pair

A, A^*	B, B^*	E, E^*	F, F^*	G, G^*
$a_6 = a_6^* = 0$	$b_6 = b_6^* = 0$	$e_9 = e_9^* = 1$	$f_{18} = f_{18}^* = 0$	$g_9 = g_9^* = 0$
$a_9 = a_9^* = 0$	$b_9 = b_9^* = 0$	$e_{13} = e_{13}^* = 1$	$f_{29} = f_{29}^* = 0$	$g_{13} = g_{13}^* = 0$
$a_{18} = a_{18}^* = 0$	$b_{18} = b_{18}^* = 0$	$e_{18} = e_{18}^* = 1$		$g_{19} = g_{19}^* = 0$
$a_{20} = a_{20}^* = 0$	$b_{20} = b_{20}^* = 0$	$e_{19} = e_{19}^* = 1$		
$a_{25} = a_{25}^* = 0$	$b_{25} = b_{25}^* = 0$	$e_{29} = e_{29}^* = 1$		
$a_{29} = a_{29}^* = 0$	$b_{29} = b_{29}^* = 0$			

Table 6. A Differential Characteristic for E^1 ($M' = \{3, 14, 15, 24, 25\}$)

Round (i)	ΔA^i	ΔB^i	ΔC^i	ΔD^i	ΔE^i	ΔF^i	ΔG^i	ΔH^i	Prob.
23	0	$e_{9,18,29}$	0	0	e_{31}	e_M	0	0	2^{-13}
24	0	0	$e_{9,18,29}$	0	0	e_{31}	e_M	0	2^{-10}
25	e_{31}	0	0	$e_{9,18,29}$	e_{31}	0	e_{31}	e_M	2^{-10}
26	0	e_{31}	0	0	0	e_{31}	0	e_{31}	2^{-2}
27	0	0	e_{31}	0	0	0	e_{31}	0	2^{-2}
28	0	0	0	e_{31}	0	0	0	e_{31}	1
29	e_{31}	0	0	0	0	0	0	0	2^{-4}
30	$e_{9,18,29}$	e_{31}	0	0	0	0	0	0	2^{-6}
31	$e_{5,27}$	$e_{9,18,29}$	e_{31}	0	0	0	0	0	2^{-12}
32	$e_{M'}$	$e_{5,27}$	$e_{9,18,29}$	e_{31}	0	0	0	0	2^{-15}
33	$e_{11,23}$	$e_{M'}$	$e_{5,27}$	$e_{9,18,29}$	e_{31}	0	0	0	·

Table 7. Number of Differential Characteristics for Rounds 23-32 (E^1) With Respect to Probabilities

Prob. (q_γ)	2^{-74}	2^{-75}	2^{-76}	2^{-77}	$\cdots$
Number of Characteristics	15	60	236	440	$\cdots$

As mentioned before, a related-key rectangle distinguisher does not use a related-key differential but a differential for E^1. Thus we need not concern difference propagations of round keys for E^1. Our differential characteristic $\gamma \rightarrow \delta$ for E^1 has 10 rounds ($23 \sim 32$) and a probability of 2^{-74}, where $\gamma = (0, e_{9,18,29}, 0, 0, e_{31}, e_{6,9,18,20,25,29}, 0, 0)$ and $\delta = (e_{11,23}, e_{3,14,15,24,25}, e_{5,27}, e_{9,18,29}, e_{31}, 0, 0, 0)$. See Table 6 for the details of this characteristic.

In order to obtain $\hat{p^*}$ (resp., $\hat{q}$) we should compute the probabilities of all the related-key differentials with input difference α through E^0 (resp., all the differentials with output difference δ through E^1). However, it is computationally infeasible to compute the probabilities of all these differentials. So we take into account as many related-key differential characteristics for E^0 (resp., many differential characteristics for E^1) as possible for computing a lower bound of $\hat{p^*}$ (resp., $\hat{q}$). In our observation, however, any 23-round related-key differential characteristic with input difference α for E^0 has a probability which is much less than 2^{-31}, so we ignore in our distinguisher the probabilities derived from all related-key differential characteristics for E^0 except for the characteristic as described in Table 4.

In order to compute a lower bound of $\hat{q}$ we settle for counting over a wide variety of differential characteristics which have the same last 9 rounds in the forgoing 10-round differential characteristic for E^1. Table 7 shows the number of counted differential characteristics according to their probabilities. As a result, we can increase a lower bound for $\hat{p^*}$ (resp., $\hat{q}$) up to 2^{-31} (resp., $2^{-71.16}$). Since the value $\hat{p^*} \cdot \hat{q} (\approx 2^{-102.16})$ is greater than 2^{-128}, this 33-round related-key rectangle distinguisher can distinguish between 33-round SHACAL-2 and a random permutation.

We now present a method to use the 33-round related-key rectangle distinguisher to find a master key pair of 37-round SHACAL-2 where the cipher uses related keys whose difference is $(0,0,0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0)$. The attack procedure is performed as follows.

1. Prepare $2^{232.16}$ plaintext pairs (P_i, P_i^*), $i = 0, 1, \cdots, 2^{232.16} - 1$ that have the α difference and the 22-bit fixed values. (Note that each P_i is encrypted using a key k, and each P_i^* is encrypted using a key k^* where k and k^* have a difference $(0,0,0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0)$.) Encrypt all these plaintext pairs to get the $2^{232.16}$ ciphertext pairs (C_i, C_i^*).
2. Guess two 128-bit subkeys $sk(= W^{33}, W^{34}, W^{35}, W^{36})$ and $sk^*(= W^{*33}, W^{*34}, W^{*35}, W^{*36})$.
3. Partially decrypt all C_i (resp., C_i^*) through rounds $36 \sim 33$ using the sk subkey (resp., the sk^* subkey), and denote the values we get by T_i (resp., T_i^*). We keep all T_i, T_i^* values in a hash table.
4. Check that $T_{i_1} \oplus T_{i_2} = T_{i_1}^* \oplus T_{i_2}^* = \delta$ for all indexes i_1, i_2 such that $0 \le i_1 < i_2 \le 2^{232.16} - 1$. If the number of quartets passing this δ test is greater than or equal to 6, then keep the guessed 128-bit subkey sk together with all the quartets $(T_{i_1}, T_{i_2}, T_{i_1}^*, T_{i_2}^*)$ which satisfy the δ test. Otherwise, go to Step 2.
5. Guess two 32-bit subkeys W^{32}, W^{*32}, and then partially decrypt all suggested values T_{i_1}, T_{i_2} (resp., $T_{i_1}^*, T_{i_2}^*$) through round 32 using the W^{32} (resp., W^{*32}) subkey and denote the values we get by T'_{i_1}, T'_{i_2} (resp., $T'^*_{i_1}, T'^*_{i_2}$). If all the T'_{i_1}, T'_{i_2} pairs and the $T'^*_{i_1}, T'^*_{i_2}$ pairs have the $(e_{3,14,15,24,25}, e_{5,27}, e_{9,18,29}, e_{31}, 0, 0, 0, 0)$ difference (which is the input difference of round 32 in our differential characteritic for E^1), keep the guessed 32-bit subkey W^{32} together with all the quartets $(T'_{i_1}, T'_{i_2}, T'^*_{i_1}, T'^*_{i_2})$ which satisfy the above difference. Otherwise, go to Step 5, i.e., restart this step, but in case 32-bit subkeys W^{32}, W^{*32} are all tested in this step, go to Step 2.
6. Guess two 32-bit subkeys W^{31}, W^{*31}, and then partially decrypt all suggested values T'_{i_1}, T'_{i_2} (resp., $T'^*_{i_1}, T'^*_{i_2}$) through round 31 using the W^{31} (resp., W^{*31}) subkey and denote the values we get by T''_{i_1}, T''_{i_2} (resp., $T''^*_{i_1}, T''^*_{i_2}$). If all the T''_{i_1}, T''_{i_2} pairs and $T''^*_{i_1}, T''^*_{i_2}$ pairs have the $(e_{5,27}, e_{9,18,29}, e_{31}, 0, 0, 0, 0, 0)$ difference (which is the input difference of round 31 in our differential characteritic for E^1), keep the guessed 32-bit subkey W^{31}. Otherwise, go to Step 6, i.e., restart this step, but in case 32-bit subkeys W^{31}, W^{*31} are all tested in this step, go to Step 2 or 5, i.e., if there is any key pair (W^{32}, W^{*32}) which is not yet tested for the suggested key pair (sk, sk^*), go to Step 5, otherwise, go to Step 2.
7. For the suggested subkey sk, W^{32}, W^{31}, do an exhaustive search for the 320-bit remaining key using trial encryption (For the suggested subkey, we use two known plaintext and ciphertext pairs for the trial encryption). If a 512-bit key k' is suggested, output the key as a master key of 37-round SHACAL-2. In this case, we also output the key $k' \oplus (0,0,0,0,0,0,0,0,e_{31},0,0,0,0,0,0,0)$ as the related master key of 37-round SHACAL-2.

The data complexity of this attack is $2^{233.16}$ related-key chosen plaintexts. The memory requirements of this attack are dominated by the memory for ciphertext pairs, so this attack requires about $2^{238.16}(= 2^{233.16} \cdot 32)$ memory bytes.

The time complexity of Step 1 (the data collection step) is $2^{233.16}$ 37-round SHACAL-2 encryptions and the time complexity of Step 3 is $2^{484.95}(\approx 2^{233.16} \cdot 2^{256} \cdot \frac{1}{2} \cdot \frac{4}{37})$ 37-round SHACAL-2 encryptions on average (The factor $\frac{1}{2}$ means the average fraction of 128-bit subkey pairs which are tested in Step 3). In Step 4, each of all possible quartets must be compared to the δ difference twice. This can be done efficiently by using a hash table for checking the δ difference (Similarly, the hash table can be used for checking the differences suggested in Step 5,6). In Step 4, the probability that each wrong subkey pair produces at least 6 quartets passing the δ test is about $2^{-59.22}(\approx \sum_{i=6}^{t} ({}_tC_i \cdot (2^{-256 \cdot 2})^i \cdot (1 - 2^{-256 \cdot 2})^{t-i}))$ where t is the value $2^{463.32}$ which represents the number of all possible quartets derived from the $2^{232.16}$ plaintext pairs. It follows that the expected number of the suggested 128-bit subkey pairs in Step 4 is $2^{195.78}(\approx 2^{256} \cdot \frac{1}{2} \cdot 2^{-59.22})$ on average.

As one of the methods for reducing the suggested $2^{195.78}$ subkey pairs, Steps 5 and 6 are performed in this attack. Additionally, we can obtain other 64-bit subkey pairs through Steps 5 and 6. The probability that each wrong subkey pair (W^{32}, W^{*32}) satisfies the requirement of Step 5 is at most $2^{-180}(= (2^{-15})^{12})$. Because a probability 2^{-15} is required to satisfy a 1-round differential characteristic for rounds $32 \sim 33$ described in Table 6 and the number of the quartets tested in Step 5 is at least 6. So the expected number of the suggested 160-bit subkey pairs $((sk, W^{32}), (sk^*, W^{*32}))$ in Step 5 is $2^{79.78}(\approx 2^{195.78} \cdot 2^{64} \cdot 2^{-180})$ and the time complexity of Step 5 is about $2^{256.16}(\approx 2^{195.78} \cdot 2^{64} \cdot 12 \cdot \frac{1}{37})$ 37-round SHACAL-2 encryptions. Similarly, Step 6 exploits a 1-round differential characteristic for rounds $31 \sim 32$ described in Table 6 for reducing the suggested $2^{79.78}$ subkey pairs. Since this 1-round differential characteristic has a probability of 2^{-12} and the number of the quartets tested in Step 6 is at least 6, the expected number of the suggested 192-bit subkey pairs $((sk, W^{32}, W^{31}), (sk^*, W^{*32}, W^{*31}))$ in Step 6 is $2^{-0.22}(\approx 2^{79.78} \cdot 2^{64} \cdot (2^{-12})^{12})$ and the time complexity of Step 6 is about $2^{142.16}(\approx 2^{79.78} \cdot 2^{64} \cdot 12 \cdot \frac{1}{37})$ 37-round SHACAL-2 encryptions. Since the time complexity of Step 7 is about 2^{320} 37-round SHACAL-2 encryptions, the total time complexity of this attack is about $2^{484.95}(\approx 2^{233.16} + 2^{484.95} + 2^{256.16} + 2^{142.16} + 2^{320})$ 37-round SHACAL-2 encryptions.

Since this attack uses the forgoing 33-round related-key rectangle distinguisher with a probability of $(\hat{p^*} \cdot \hat{q})^2(\approx (2^{-102.16})^2)$, the expected number of right quartets is about $2^3(= {}_{2^{232.16}}C_2 \cdot 2^{-256} \cdot (2^{-102.16})^2)$. It follows that the probability that a right subkey pair satisfies the requirement of Steps 5 and 6 is one. Thus, the success rate of this attack, i.e., the probability which the right subkey pair produces at least 6 quartets passing the δ test is about $0.80(\approx \sum_{i=6}^{t} ({}_tC_i \cdot (2^{-256} \cdot (2^{-102.16})^2)^i \cdot (1 - 2^{-256} \cdot (2^{-102.16})^2)^{t-i}))$ where t is the value $2^{463.32}$.

6 Conclusion

In this paper we have presented the enhanced related-key differential-linear attack and we have extended it into the related-key differential-nonlinear attack. Using the related-key differential-nonlinear attack we have broken 35-round SHACAL-2 with a data complexity of $2^{42.32}$ related-key chosen plaintexts and a time complexity of $2^{452.1}$ encryptions. We have also presented a related-key rectangle attack on 37-round SHACAL-2. Our attack requires $2^{233.16}$ related-key chosen plaintexts and $2^{484.95}$ encryptions.

We believe that the methods developed to attack SHACAL-2 can be efficiently applied to block ciphers with weak key scheduling algorithms.

References

1. E. Biham and A. Shamir, *Differential cryptanalysis of the full 16-round DES*, Advances in Cryptology - CRYPTO 1992, LNCS 740, pp. 487-496, Springer-Verlag, 1992.
2. E. Biham, *New Types of Cryptanalytic Attacks Using Related Keys*, Journal of Crytology, v. 7, n. 4, pp.229-246, 1994.
3. E. Biham, O. Dunkelman and N. Keller, *Enhanced Differential-Linear Cryptanalysis*, Advances in Cryptology - ASIACRYPT 2002, LNCS 2501, pp. 254-266, Springer-Verlag, 2002.
4. E. Biham, O. Dunkelman and N. Keller, *Rectangle Attacks on 49-Round SHACAL-1*, FSE 2003, LNCS 2887, pp. 22-35, Springer-Verlag, 2003.
5. H. Handschuh and D. Naccache, *SHACAL : A Family of Block Ciphers*, Submission to the NESSIE project, 2002.
6. H. Handschuh, L.R. Knudsen and M.J. Robshaw, *Analysis of SHA-1 in Encryption Mode*, CT-RSA 2001, LNCS 2020, pp. 70-83, Springer-Verlag, 2001.
7. P. Hawkes, *Differential-Linear Weak-Key Classes of IDEA*, Advances in Cryptology - EUROCRYPT 1998, LNCS 1403, pp. 112-126, Springer-Verlag, 1998.
8. S. Hong, J. Kim, G. Kim, J. Sung, C. Lee and S. Lee, *Impossible Differential Attack on 30-Round SHACAL-2*, INDOCRYT 2003, LNCS 2904, pp. 97-106, Springer-Verlag, 2003.
9. J. Kim, D. Moon, W. Lee, S. Hong, S. Lee and S. Jung, *Amplified Boomerang Attack against Reduced-Round SHACAL*, Advances in Cryptology - ASIACRYPT 2002, LNCS 2501, pp. 243-253, Springer-Verlag, 2002.
10. J. Kim, G. Kim, S. Hong, S. Lee and D. Hong, *The Related-Key Rectangle Attack - Application to SHACAL-1*, ACISP 2004, To appear.
11. L.R. Knudsen, *Trucated and Higher Order Differentials*, FSE 1996, LNCS 1039, pp 196-211, Springer-Verlag, 1995.
12. S.K. Langford and M.E. Hellman, *Differential-Linear Cryptanalysis*, Advances in Cryptology - CRYPTO 1994, LNCS 839, pp. 17-25, Springer-Verlag, 1994.
13. M. Matsui, *Linear Cryptanalysis Method for DES Cipher*, Advances in Cryptology - EUROCRYPT 1993, LNCS 765, pp. 386-397, Springer-Verlag, 1994.
14. A.A. Selcuk, A. Bicak, *On Probability of Success in Linear and Differential Cryptanalysis*, SCN 2002, LNCS 2576, pp. 174-185, Springer-Verlag, 2002.
15. Y. Shin, J. Kim, G. Kim, S. Hong and S. Lee, *Differential-Linear Type Attacks on Reduced Rounds of SHACAL-2*, ACISP 2004, To appear.

16. U.S. Department of Commerce.*FIPS 180-1*: Secure Hash Standard ,Federal Information Processing Standards Publication, N.I.S.T., April 1995.
17. U.S. Department of Commerce.*FIPS 180-2*: Secure Hash Standard ,Federal Information Processing Standards Publication, N.I.S.T., August 2002.

Related-Key Attacks on DDP Based Ciphers: CIKS-128 and CIKS-128H*

Youngdai Ko[1], Changhoon Lee[2], Seokhie Hong[2], Jaechul Sung[3], and Sangjin Lee[2]

[1] Information Security Team, LG CNS, Hoehyeon Dong , Jung Gu, Seoul, Korea
koyd@lgcns.com
[2] Center for Information Security Technologies(CIST), Korea University, Anam Dong, Sungbuk Gu, Seoul, Korea
{crypto77, hsh, sangjin}@cist.korea.ac.kr
[3] Department of Mathematics, University of Seoul, 90 Cheonnong Dong, Dongdaemun Gu, Seoul, 130-743, Korea
jcsung@uos.ac.kr

Abstract. CIKS-128 and CIKS-128H are 128-bit block ciphers with a 256-bit key sizes based on data-dependent operations, respectively. They are also fast hardware-oriented ciphers and improvements of block cipher CIKS-1 introduced in [14]. This paper presents related-key differential attacks on full-round CIKS-128 and CIKS-128H. In result, using full-round related-key differential characteristics with probability 2^{-36} and $2^{-35.4}$, these attacks can recover the partial subkey bits for CIKS-128 and CIKS-128H with about 2^{40} plaintexts, respectively. These works suggests that the greatest possible care has to be taken when proposing improvements of the existing block ciphers.

Keywords: CIKS-128, CIKS-128H, Block Cipher, Related-Key Differential Attack, Data-Dependent Operation.

1 Introduction

Data dependent operations (DDO) are very attractive cryptographic primitive designing fast block ciphers suitable for the applications of many network requiring high speed encryption [15, 16, 17]. Up to now, DDO based ciphers [4, 14] seem to be secure against well known attack methods like differential cryptanalysis and linear cryptanalysis [1, 13, 12, 9, 3]. CIKS-128 [2] and CIKS-128H [16] are such ciphers based on DDO designed by N.D.Goots et al. and N.Sklavos et al. respectively. As CIKS-128H is just a modified version of CIKS-128, it use a different elementary control box performing DDO from CIKS-128. These algorithms have very simple key schedules for the case of frequent change of keys. However,

* This work was supported by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 191–205, 2004.

such simple key schedules may be vulnerable from the view point of related-key cryptanalysis. It is known that related-key cryptanalysis is a practical attack on key-exchange protocols that do not guarantee key integrity, and key-update protocols that updates session keys using a known function, for example, K, $K+1$, $K+2$, etc. where K is a session key [5, 6].

In this paper, we present the related-key differential attacks on full-round CIKS-128 and CIKS-128H. To begin with, we derive the properties of non-linear function G and DDO-boxes used in round function of CIKS-128 and CIKS-128H, which allow us to exploit full-round related-key differential characteristics. Then, we construct a full-round related-key differential characteristic with probability 2^{-36} and $2^{-35.4}$ for CIKS-128 and CIKS-128H, respectively, and recover the partial subkey bits of CIKS-128 and CIKS-128H with about 2^{40} plaintexts, respectively. Our attacks do not need the extra encryption procedures in order to recover the partial subkey bits. We only need to observe the difference propagation of ciphertext pair maintaining our related-key differential characteristics of full-round CIKS-128 and CIKS-128H, and trace the corresponding subkey bits, respectively.

This paper is organized as follows; In Section 2, we mention notations used in this paper and introduce several properties of DDO-boxes. Section 3 shortly describes algorithms and properties of CIKS-128 and CIKS-128H. We present the related-key differential attacks of CIKS-128 and CIKS-128H in Section 4 and conclude in Section 5.

2 Preliminaries

In this section, we shortly describe various $P_{n/m}$ and $R_{n/m}$-boxes executing the data dependent operations and present their several properties. For convenience, we use the same notations represented in [2, 16].

2.1 Notations

We will use the following notations throughout this paper. Note that bits will be numbered from left to right, starting at bit 1. For example, the right most bit of a n-bit binary string P is p_n, while the leftmost bit is p_1, i.e., $P = (p_1, p_2, \cdots, p_n)$.

- $P_{low} = (p_1, p_2, \cdots, p_{\frac{n}{2}})$, $P_{hi} = (p_{\frac{n}{2}+1}, p_{\frac{n}{2}+2}, \cdots, p_n)$
- α_i : a 4-bit binary string in which the i-th bit is one and the others are zero
- β_i : a 8-bit binary string in which the i-th bit is one and the others are zero
- e_i : a 64-bit binary string in which the i-th bit is one and the others are zero
- d_i : a 192-bit binary string in which the i-th bit is one and the others are zero
- $\oplus$: a bitwise-XOR operation
- $\otimes$: a bitwise-AND operation
- $|$: a concatenation operation
- $\lll$: a left cyclic rotation

Table 1. The results of (a) $P_{2/1(v)}(x_1x_2)$ and (b) $R_{2/1(v)}(x_1x_2)$, (where $x_1, x_2 \in \{0,1\}$

$x_1\,x_2$		00	01	10	11
$P_{2/1(v)}$	v=0	00	01	10	11
	v=1	00	10	01	11

(a) $P_{2/1(v)}(x_1\,x_2)$

$x_1\,x_2$		00	01	10	11
$R_{2/1(v)}$	v=0	00	10	01	11
	v=1	00	01	11	10

(b) $R_{2/1(v)}(x_1\,x_2)$

2.2 DDO-Boxes

DDO-boxes $P_{m/n}$ and $R_{m/n}$ used in CIKS-128 and CIKS-128H are constructed as a superposition of standard elementary controlled boxes $P_{2/1}$- and $R_{2/1}$-box, respectively(see Fig. 1,2). In other words, $P_{m/n}$(resp. $R_{m/n}$) are constructed by using $P_{2/1}$(resp. $R_{2/1}$) repeatedly. Thus, the only difference between $P_{n/m}$ and $R_{n/m}$ is the used standard elementary controlled boxes. Here, n denotes the bit-

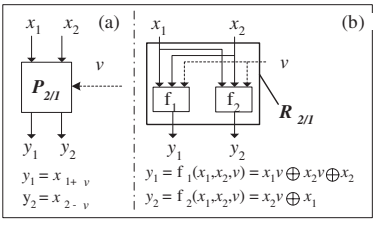

Fig. 1. Elementary controlled boxes : (a) $P_{2/1}$-box, (b) $R_{2/1}$-box

size of input and output string and m means that of control vectors. Note that $R_{2/1}$-box was designed for thwarting the linearity of $P_{2/1}$-box that the sum of input bits is equal to that of output bits [16].

Now, we present some properties of DDO-boxes. Table 1 describes two output bits of $P_{2/1}$ and $R_{2/1}$-box corresponding their two input bits under each one-bit control vector. We let x_1x_2 be two-bit input string of $P_{2/1}$ or $R_{2/1}$-box and v be one-bit control vector, where x_1, x_2 and $v \in \{0,1\}$.

Property 1. $P_{2/1(0)}(x_1x_2) = P_{2/1(1)}(x_1x_2)$ with a probability of 2^{-1}.

Property 2. $R_{2/1(0)}(x_1x_2) = R_{2/1(1)}(x_1x_2)$ with a probability of 2^{-2}.

Property 1 holds when $x_1 = x_2$ and *Property* 2 holds when $x_1 = x_2 = 0$.

Property 3. Let V and V' be m-bit control vectors for $P_{n/m}$-box and assume $V \oplus V' = d_i$ $(1 \le i \le m)$, then $P_{n/m(V)}(X) = P_{n/m(V')}(X)$ with a probability of 2^{-1} where $X \in \{0,1\}^n$. It is also held in the inverse of $P_{n/m}$-box, i.e., $P^{-1}_{n/m}$-box.

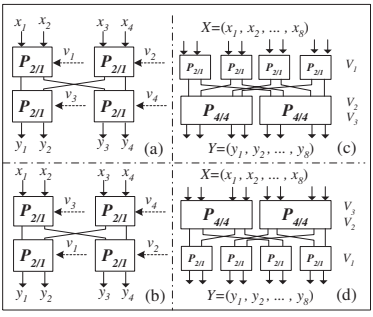

Fig. 2. Examples of DDO-boxes : (a) $P_{4/4}$, (b) $P^{-1}_{4/4}$, (c) $P_{8/12}$, (b) $P^{-1}_{8/12}$

Property 4. Let V and V' be m-bit control vectors for $R_{n/m}$-box and assume $V \oplus V' = d_i$ $(1 \le i \le m)$, then $R_{n/m(V)}(X) = R_{n/m(V')}(X)$ with a probability of 2^{-2} where $X \in \{0,1\}^n$. It is also held in the inverse of $R_{n/m}$-box, i.e., $R^{-1}_{n/m}$-box.

Property 5. If $X \oplus X' = \alpha_i$ and $W \oplus W' = \beta_k$, then $P_{4/4(V)}(X) \oplus P_{4/4(V)}(X')$ $= \alpha_j$ and $P_{8/12(V')}(W) \oplus P_{8/12(V')}(W') = \beta_h$, where $X, X', V \in \{0,1\}^4$, $W, W' \in \{0,1\}^8$, $V' \in \{0,1\}^{12}$, $1 \le i, j \le 4$, $1 \le k, h \le 8$. In addition, if (i, j) and (k, h) are fixed, then the exact route from i to j and from k to h, via respective 2 and 3 elements ($P_{2/1}$-boxes) are also fixed.

Property 6. If we assume that $x'_1 = x_1 \oplus 1$ and $x'_2 = x_2 \oplus 1$, we know the following facts.

$$R_{2/1(0)}(x_1x_2) \oplus R_{2/1(0)}(x_1x'_2) = 10$$
$$R_{2/1(1)}(x_1x_2) \oplus R_{2/1(1)}(x_1x'_2) = 01.$$
$$R_{2/1(0)}(x_1x_2) \oplus R_{2/1(0)}(x'_1x_2) = 01$$
$$R_{2/1(1)}(x_1x_2) \oplus R_{2/1(1)}(x'_1x_2) = 11.$$

Property 7. If $X \oplus X' = \alpha_4$ and $W \oplus W' = \beta_8$, then by *Property* 6, $R_{4/4(V)}(X)$ $\oplus$ $R_{4/4(V)}$ $(X') = \alpha_j$ and $R_{8/12(V')}(W) \oplus R_{8/12(V')}(W') = \beta_h$, where $X, X', V \in \{0,1\}^4$, $W, W' \in \{0,1\}^8$, $V' \in \{0,1\}^{12}$, $1 \le j \le 4$, $1 \le h \le 8$. In addition, if j and h are fixed, then then the exact route from 4 to j and from 8 to h, via respective 2 and 3 elements ($R_{2/1}$-boxes) are also fixed.

Property 8. If $W \oplus W' = \beta_k$, then $P^{-1}_{8/12(V')}(W) \oplus P^{-1}_{8/12(V')}(W') = \beta_h$, where W, W' $\in \{0,1\}^8$, $V' \in \{0,1\}^{12}$, $1 \le k, h \le 8$. Particularly, if $k \in \{1,2,3,4\}$ then h is one of $1, 2, 5, 6$ and if $k \in \{5,6,7,8\}$ then h is one of $3, 4, 7, 8$. In addition, if k and h are fixed, there are only two possible routes from k to h, via 3 elements ($P_{2/1}$-boxes).

Property 9. If $W \oplus W' = \beta_8$, then $R^{-1}_{8/12(V')}(W) \oplus R^{-1}_{8/12(V')}(W') = \beta$, where $\beta \in$ $\{\beta_{3,4}, \beta_4, \beta_{7,8}, \beta_8\}$, $W, W' \in \{0,1\}^8$, $V' \in \{0,1\}^{12}$, $1 \le k, h \le 8$. By the properties of the structure of $R^{-1}_{8/12}$-box and *Property* 6, if $\beta \in \{\beta_{3,4}, \beta_{7,8}\}$ then the route from 8 to 3,4 or 7,8 is fixed and if $\beta \in \{\beta_4, \beta_8\}$ then, there are only two possible routes from 8 to 4 or 8, via 3 elements ($R_{2/1}$-boxes) in each case.

Property 10. Let $Y = P_{n/m(V)}(X)$ and $Y' = P_{n/m(V)}(X')$. Then $Hw(X \oplus X')$=$Hw(Y \oplus Y')$. Similary, the inverse is also held.

Property 7, 8 and 9 mean that if we know the bit-position of input vector which has a difference and corresponding that of output vector, then we can determine the route of difference, i.e., we can derive the control vector of corresponding to the elements of $P_{n/m}$ or $R_{n/m}$-box [1]. For example, in *Property* 5 let $k = 8$ and $h = 2$. Then, we can exactly know the 3 bits control vector of corresponding to the elements ($P_{2/1}$-boxes) of $P_{8/12}$-box with probability 1. See Fig. 3. The bold line denotes the propagation of the right most bit which has a difference. *Property* 10 describes the linearity of DDO-boxes based on $P_{2/1}$.

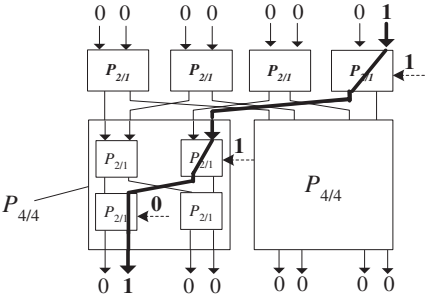

Fig. 3. An example of the difference propagation and the corresponding 3 bits controlled vector in $P_{8/12}$-box

3 CIKS-128 and CIKS-128H

In this section, we briefly describe CIKS-128 and CIKS-128H respectively. For details, refer to [2, 16].

3.1 Description of CIKS-128

CIKS-128 is a 12-round iterated block with 128-bit block size and 256-bit key size. The general encryption scheme is composed of the initial key XOR operation, round function *Crypt* and the final key XOR operation. The round function *Crypt* uses various operations such as DDO-boxes($P_{128/1}$, $P_{64/192}$ and $P^{-1}_{64/192}$), non-linear function G, fixed permutations (Π, π and I), and XOR operations. For details, refer to [2] and Fig. 4.

The key schedule of CIKS-128 is very simple. The 256-bit master key K is split into four 64-bit blocks ,i.e., $K = (K_1, K_2, K_3, K_4)$. Then, in order to generate the subkey sequence $A^{(r)} = (A_1^{(r)}, A_2^{(r)}, A_3^{(r)}, A_4^{(r)})$, K_1,K_2,K_3 and K_4 are

[1] As we mentioned in 2.2, DDO-box can be considered as a superposition of the standard elementary $P_{2/1}$-boxes.

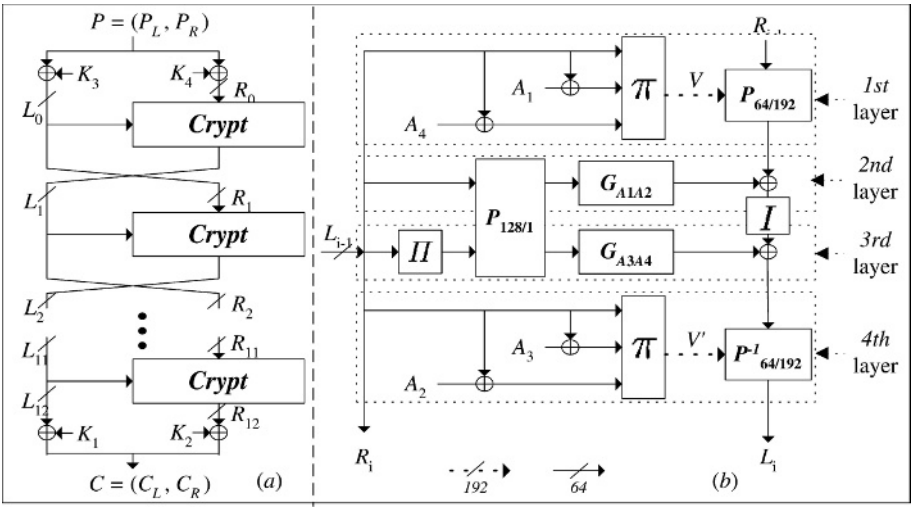

Fig. 4. (a) General structure of CIKS-128 and (b) round function *Crypt*

rearranged as specified in Table Here $A^{(r)}$ denotes the r-th round key sequence, and $A_i^{(r)}, K_j \in \{0,1\}^{64}$, $1 \leq i, j \leq 4$ and $1 \leq r \leq 12$. 2.

Table 2. Key schedule of CIKS-128

r	1	2	3	4	5	6	7	8	9	10	11	12
A_1	K_1	K_4	K_3	K_2	K_1	K_3	K_3	K_1	K_2	K_3	K_4	K_1
A_2	K_2	K_3	K_4	K_1	K_2	K_4	K_4	K_2	K_1	K_4	K_3	K_2
A_3	K_3	K_2	K_1	K_4	K_3	K_1	K_1	K_3	K_4	K_1	K_2	K_3
A_4	K_4	K_1	K_2	K_3	K_4	K_2	K_2	K_4	K_3	K_2	K_1	K_4

3.2 Description of CIKS-128H

As CIKS-128H is a modified version of CIKS-128, it is almost similar to CIKS-128. The differences between CIKS-128 and CIKS-128H are DDO-box and key schedule. CIKS-128H replaced $P_{64/192}$ and $P_{64/192}^{-1}$-boxes in CIKS-128 by $R_{64/192}$ and $R_{64/192}^{-1}$-boxes which are composed of the elementary controlled boxes $R_{2/1}$. The structure of round function of CIKS-128H is equal to that of CIKS-128 except operations $R_{64/192}$ and $R_{64/192}^{-1}$-boxes. In addition, CIKS-128H uses a little different key schedule from CIKS-128 for 6-round or 8-round represented in Table 3. Here, we deal with 8-round CIKS-128H. For details, refer to [16].

Table 3. Key schedule of 8-round CIKS-128H

r	1	2	3	4	5	6	7	8
A_1	K_1	K_4	K_3	K_2	K_2	K_3	K_4	K_1
A_2	K_2	K_3	K_4	K_1	K_1	K_4	K_3	K_2
A_3	K_3	K_2	K_1	K_4	K_4	K_1	K_2	K_3
A_4	K_4	K_1	K_2	K_3	K_3	K_2	K_1	K_4

3.3 Properties of CIKS-128 and CIKS-128H

In this subsection, we derive some properties of operations used in round function of CIKS-128 and CIKS-128H, which are useful to construct related key differential characteristics of of CIKS-128 and CIKS-128H.

Property 11. i) For the control vector V of $P_{64/192}$-box, $\pi(L, A', A'') \oplus \pi(L, A' \oplus e_{64}, A'') = d_{138}$ and $\pi(L, A', A'') \oplus \pi(L, A', A'' \oplus e_{64}) = d_{180}$. ii) for the control vector V' of $P^{-1}_{64/192}$-box, $\pi(L, A', A'') \oplus \pi(L, A' \oplus e_{64}, A'') = d_{42}$ and $\pi(L, A', A'') \oplus \pi(L, A', A'' \oplus e_{64}) = d_{20}$. where $L, A', A'' \in \{0,1\}^{64}$ and $V, V' \in \{0,1\}^{192}$

Property 12. $G(X, A', A'') \oplus G(X, A' \oplus e_{64}, A'' \oplus e_{64}) = 0$, or
$G(X, A', A'') \oplus G(X, A' \oplus e_{64}, A'' \oplus e_{64}) = e_{64}$.

Property 13. Let $Y = P^{-1}_{64/192(V)}(X)$ and $Y' = P^{-1}_{64/192(V)}(X \oplus e_i)$. Then, $Y \oplus Y' = e_j$ by *Property* 10 and there are exactly two possible routes from i to j by *Property* 5 and 8.

Property 14. Let $X \oplus X' = e_{64}$, $Y = R^{-1}_{64/192(V)}(X)$ and $Y' = R^{-1}_{64/192(V)}(X')$. If $Y \oplus Y' = e_i$ or $e_{i,i+1}$, then there are exactly two routes from the 64-th bit position (of input difference) to the i-th or i-th and $(i+1)$-th bit position (of output difference) by *Property* 6, 7 and 9, where $i = 4j-1$, $1 \le j \le 16$.

4 Related-Key Differential Attacks on Full-Round CIKS-128 and CIKS-128H

In this section, we construct a related-key differential characteristic for CIKS-128 and CIKS-128H using the properties mentioned in Subsection 2.2 and 3.3 and present the related-key differential attacks on full-round CIKS-128 and CIKS-128H.

4.1 Related-Key Differential Characteristic of CIKS-128

We survey related-key differential properties for round function *Crypt*. We consider the situation that two identical input values of *Crypt* change into two

identical output values under the condition that the corresponding round keys which have a relation. To begin with, we suppose there exist two round keys $\mathsf{A}' = (A'_1, A'_2, A'_3, A'_4)$ and $\mathsf{A}'' = (A''_1, A''_2, A''_3, A''_4)$ as follows:

$$\mathsf{A}' \oplus \mathsf{A}'' = (0, 0, e_{64}, e_{64}) \text{ or } (e_{64}, e_{64}, 0, 0),$$

where $\mathsf{A}', \mathsf{A}'' \in \{0,1\}^{256}$, $A'_i, A''_i \in \{0,1\}^{64}$ and $1 \leq i \leq 4$.

We call the subkey sequences A'_i and A''_i related on e_{64} 'the related subkey sequence $\mathcal{A}_i = (A'_i, A''_i)$' and shortly denote it by $\mathcal{A}_i$, e.g., in $\mathsf{A}' \oplus \mathsf{A}'' = (0,0,e_{64},e_{64})$ the related subkey sequences are $\mathcal{A}_3$ and $\mathcal{A}_4$. Thus, according to the condition of $\mathsf{A}' \oplus \mathsf{A}''$, we classify them into two cases *C1.* and *C2.* as follows. In addition, for convenience, we divide the round function *Crypt* into four layers. Refer to Fig. 4 (b).

C1. $\mathsf{A}' \oplus \mathsf{A}'' = (0, 0, e_{64}, e_{64})$
This case means that the related subkey sequences are $\mathcal{A}_3$ and $\mathcal{A}_4$. In the 1st layer, $\mathcal{A}_4$ is used to form the 192-bit control vector V for the $P_{64/192}$-box. In the 4-th layer, $\mathcal{A}_3$ is used to form the control vector V' for the $P^{-1}_{64/192}$-box. $\mathcal{A}_3$ and $\mathcal{A}_4$ are also used of subkey sequences of G produced a 64-bit output value XORed with a result of permutation I in the 3rd layer (see Fig. 4 (b)). Thus, by *Property* 11 and 3, if the input difference of the 1st layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Similarly, if the input difference of the 4-th layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Furthermore, by *Property* 12, if the input difference of the 3rd layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Hence, if Crypt have two identical inputs and the corresponding round keys satisfying the case *C1.* then Crypt outputs the same values each other with a probability of 2^{-3}.

C2. $\mathsf{A}' \oplus \mathsf{A}'' = (e_{64}, e_{64}, 0, 0)$
This case is similar to *C1.*. It means that the related subkey sequences are $\mathcal{A}_1$ and $\mathcal{A}_2$. In the 1st layer, $\mathcal{A}_1$ is used to form the 192-bit controlled vector V for the $P_{64/192}$-box. In the 4-th layer, $\mathcal{A}_2$ is used to form the control vector V' for the $P^{-1}_{64/192}$-box. $\mathcal{A}_1$ and $\mathcal{A}_2$ are used of subkey sequences of G produced a 64-bit output value XORed with an output of the 1st layer in the 2nd layer. Thus, by *Property* 11 and 3, if the input difference of the 1st layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Similarly, if the input difference of the 4-th layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Furthermore, by *Property* 12, if the input difference of the 2nd layer is zero then the corresponding output difference is zero with a probability of 2^{-1}. Hence, if Crypt have two identical inputs and the corresponding round keys satisfying the case *C2.* then Crypt outputs the same values each other with a probability of 2^{-3}.

That is, these cases mean a 1-round related-key differential characteristic of CIKS-128 with a probability of 2^{-3}.

Table 4. A full-round related-key differential characteristic of CIKS-128

Round	1	2	3	4	5	6	7	8	9	10	11	12
Case	*C2*	*C1*	*C1*	*C2*	*C2*	*C1*	*C1*	*C2*	*C2*	*C1*	*C1*	*C2′*

4.2 Related-Key Differential Attack on Full-Round CIKS-128

Now, we present a related-key differential attack on full-round CIKS-128. First, we construct a full-round related key differential characteristic using the 1-round related-key differential characteristic of CIKS-128 mentioned above. In order to achieve this end, we encrypt the plaintext $P = (P_L, P_R)$ under the master keys $K = (K_1, K_2, K_3, K_4)$ and $K' = (K_1, K_2, K_3 \oplus e_{64}, K_4 \oplus e_{64})$, respectively, and then get the corresponding ciphertexts $C = (C_L, C_R)$ and $C' = (C'_L, C'_R)$, i.e., $E_K(P) = C$ and $E_{K'}(P) = C'$, where E is a block cipher CIKS-128. Then, from the key schedule of CIKS-128, we know that the relation of each round key is satisfied with *C1* or *C2*, i.e., the related subkey sequences of every round are $\mathcal{A}_1$ and $\mathcal{A}_2$, (or $\mathcal{A}_3$ and $\mathcal{A}_4$).

Table 4 shows a full-round related key differential characteristic of CIKS-128 with a probability of $(2^{-3})^{12} (= 2^{-36})$. We remark that if the input difference of non-linear function G is zero and the used subkey sequences A_1, A_2 (or A_3, A_4) are related with e_{64} respectively, then the output difference is zero with a probability 2^{-1} and it is also e_{64} with a probability of 2^{-1} (see *Property* 12). Thus, we consider the output difference $e_{64} (\neq 0)$ of the 3rd layer in the final round in order to recover the partial subkey bits of CIKS-128 in our full-round related key differential characteristic. We denote the case of final round by $C2'$.

As a result, if we consider the respective encryptions of P under keys K and $\mathsf{K'}$, then the output difference of 3rd layer in final round is e_{64}, i.e, the input difference of the $P^{-1}_{64/192}$-box is e_{64} with a probability of 2^{-35}. In addition, we do not need to consider the difference of control vector,i.e., d_{42} for $P^{-1}_{64/192}$-box in the 4-th layer of final round, because it doesn't affect the possible routes of the input difference due to the structure of $P^{-1}_{64/192}$-box. See Fig. 5.

However, in order to explain easily our attack scenario, we consider the output difference of $P_{2/1}$-box corresponding to the 42-th bit position of control vector as zero with an additional probability of 2^{-1} (See Fig. 6) [2]. That is, the difference of ciphertext pair is $(0, e_j)$ with a probability of 2^{-36} in our related-key differential characteristic. In Fig. 6, every dotted lines represent a stream of zero difference(0) and every bold line denotes that of non-zero difference(1).

Now, we explain how to recover the partial subkey bits of CIKS-128 using the above full-round related-key differential characteristic. Note that by *Property*

[2] In Fig. 6, every white box denotes an elementary $P_{2/1}$-box of $P^{-1}_{64/192}$-box which has a same control vector, i.e., the difference of control vector is zero, whereas the gray box denotes an $P_{2/1}$-box which has a different control vector, i.e., the control vector of the gray box is corresponding to the 42-th bit position of V'. In dotted boxes, we can obtain the partial subkey information.

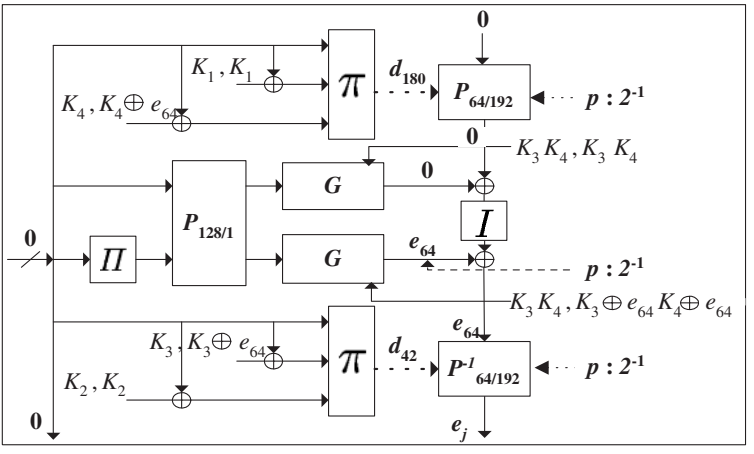

Fig. 5. Propagation of the difference in the final round

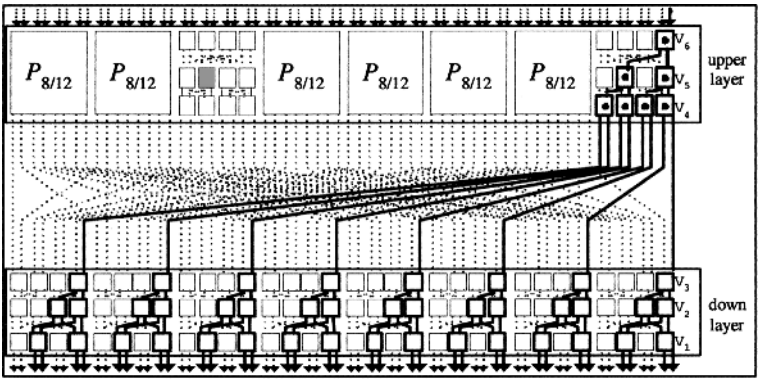

Fig. 6. The possible routes of the input difference e_{64} for the $P^{-1}_{64/192}$-box

13, if we get a ciphertext pair $C = (C_L, C_R)$ and $C' = (C_L, C_R \oplus e_j)$ such that $C \oplus C' = (0, e_j)$, then we can expect that there are only two routes[3] from the 64-th bit-position (input difference) to the j-th bit position (output difference) and also check the exact control vectors of $P_{2/1}$-boxes located in possible routes in Fig. 6. More specifically, since j is fixed, the control vector of upper layer of $P^{-1}_{64/192}$-box is uniquely determined by *Property* 5. However, in the lower layer of $P^{-1}_{64/192}$-box, there are two routes until the j-th bit position by *Property* 8, such that there exist two control vectors corresponding to each route.

[3] It is caused by the inner $P^{-1}_{8/12}$-box of $P^{-1}_{64/192}$-box, not $P_{8/12}$-box. So, for convenience, we divide the $P^{-1}_{64/192}$-box into upper layer ($P_{8/12}$-boxes) and lower layer ($P^{-1}_{8/12}$-boxes).

Table 5. Classes of difference e_j and the fixed value of control vector (v'', v', v) corresponding to the bit-position (B·P) of of upper layer in $P^{-1}_{64/192}$-box

Class	difference e_j	B·P of upper layer	(v'', v', v)
$\mathcal{CL}_1$	e_3, e_4, e_7, e_8	(44″,52′,61)	(1,1,1)
$\mathcal{CL}_2$	e_{11}, e_{12}, e_{15}, e_{16}	(44″,52′,61)	(1,1,0)
$\mathcal{CL}_3$	e_{19}, e_{20}, e_{23}, e_{24}	(44″,52′,62)	(1,0,1)
$\mathcal{CL}_4$	e_{27}, e_{28}, e_{31}, e_{32}	(44″,52′,62)	(1,0,0)
$\mathcal{CL}_5$	e_{35}, e_{36}, e_{39}, e_{40}	(44″,54′,63)	(0,1,1)
$\mathcal{CL}_6$	e_{43}, e_{44}, e_{47}, e_{48}	(44″,54′,63)	(0,1,0)
$\mathcal{CL}_7$	e_{51}, e_{52}, e_{55}, e_{56}	(44″,54′,64)	(0,0,1)
$\mathcal{CL}_7$	e_{59}, e_{60}, e_{63}, e_{64}	(44″,54′,64)	(0,0,0)

We also remark that due to the structure of $P^{-1}_{64/192}$-box, e_j is one of e_3, e_4, e_7, e_8, e_{11}, e_{12}, $\cdots$, e_{59}, e_{60}, e_{63}, e_{64}. We classify them into eight classes according to the unique control vector of upper layer like Table 5. Here, we let $l''=c_{L_l} \oplus k^1_l \oplus k^2_l$, $m'=c_{L_m} \oplus k^1_m \oplus k^3_m$, $n=c_{L_n} \oplus k^1_n$ where $c_{L_k} \in C_L$, $k^1_k \in K_1$, $k^2_k \in K_2$, $k^3_k \in K_3$ and $1 \le l, m, n, k \le 64$(Refer to Fig. 6). That is, with above observations we can obtain the partial subkey information related corresponding to the control vectors using the ciphertext pair. In this attack, we need two ciphertext pairs included to $\mathcal{CL}_i$ $(1 \le i \le 4)$, in order to obtain 8 bits subkey information.

For example, let j be 4, i.e., the difference of ciphertext pair is $(0,e_4)$. Then, there exist two routes of the difference propagation and their relation between the bit-positions and control vectors are as follows;

$$(44'', 52', 61, 16'', 25', 32) = (1,1,1,1,1,1),\ (44'', 52', 61, 16'', 26', 32) = (1,1,1,0,1,0).$$

In upper layer, the control vector of bit-position $(44'', 52', 61)$ is fixed as (1,1,1). Therefore, we can determine the exact values of $k^1_{44} \oplus k^2_{44}, k^1_{52} \oplus k^3_{52}$ and k^1_{61} because we already get $c_{L_{44}}$, $c_{L_{52}}$ and $c_{L_{61}}$. In lower layer, if we consider a set $S = \{(a,b,c,d) | a,b,c,d \in \{0,1\}\}$ as the possible vectors of bit-position $(16'', 25', 26', 61)$, there remain only four elements of S as a candidate of $(16'', 25'$,$26', 61)$ as follows; (a):(1,1,0,1) or (1,1,1,1), (b):(0,0,1,0) or (0,1,1,0). See Fig. 7.

Therefore, we can expect that there exist a right value of $k^1_{16} \oplus k^2_{16}, k^1_{25} \oplus k^3_{25}, k^1_{26} \oplus k^3_{26}$ and k^1_{32} with a probability 1/4 because we already get $c_{L_{16}}$, $c_{L_{25}}$, $c_{L_{26}}$ and $c_{L_{32}}$. Thus, if we get another ciphertexs pair $C^* = (C^*_L, C^*_R)$ and $C^{*\prime} = (C^*_L, C^*_R \oplus e_4)$, i.e., $C^* \oplus C^{*\prime} = (0, e_4)$[4] and assume that C^* and $C^{*\prime}$ are random bits strings, then we can uniquely determine the above subkey bits information. Also, in Fig. 7, we can get the value of k^1 for free, because the control vectors of bit-position 16″, 25′ (or 16″, 26′) are determined. As a result, in this example, we can find the 3 bits subkey (k^1_4,k^1_{32},k^1_{61}) and obtain the 5 bits subkey information $(k^1_{16} \oplus k^2_{16}, k^1_{25} \oplus k^3_{25}, k^1_{26} \oplus k^3_{26}, k^1_{44} \oplus k^2_{44}, k^1_{52} \oplus k^3_{52})$.

[4] It will be also enough that $C^* = (C^*_L, C^*_R)$ and $C^{*\prime} = (C^*_L, C^*_R \oplus e_3)$, i.e., $C^* \oplus C^{*\prime} = (0, e_3)$.

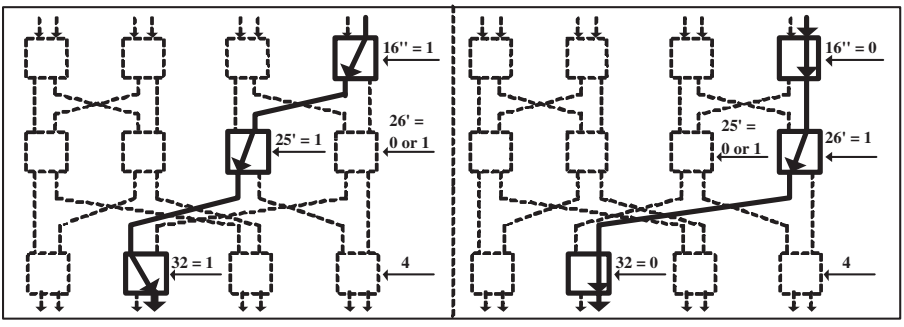

Fig. 7. Control vectors of lower layer ($P_{8/12}^{-1}$-boxes) when the output difference is e_4

Algorithm 1 describes how to get the total 47 bits subkey information. We consider 2^{41} plaintexts P_j, where $1 \leq j \leq 2^{41}$. Let $C_j = (C_{L_j}, C_{R_j})$ and $C'_j = (C'_{L_j}, C'_{R_j})$ be the ciphertexts of P_j under the keys $K = (K_1, K_2, K_3, K_4)$ and $K' = (K_1, K_2, K_3 \oplus e_{64}, K_4 \oplus e_{64})$, respectively. For each pair C_j, C'_j satisfying $C_j \oplus C'_j \in \mathcal{CL}_i$ $(1 \leq i \leq 4)$, let $(v''_{j1_{up}}, v'_{j2_{up}}, v_{j3_{up}})$ be the 3 bits control vector and k_{j1}, k_{j2}, k_{j3} be a possible subkey information of the $(v''_{j1_{up}}, v'_{j2_{up}}, v_{j3_{up}})$. We also let $(v''_{j4_{dw}}, v'_{j5_{dw}}, v'_{j6_{dw}}, v_{j7_{dw}})$ be a 4 bits control vector related on lower layer of $P_{64/192}^{-1}$-box in the final round and $k_{j4}, k_{j5}, k_{j6}, k_{j7}$ be a possible subkey information of the $(v''_{j4_{dw}}, v'_{j5_{dw}}, v'_{j6_{dw}}, v_{j7_{dw}})$.

If we get a ciphertext pair $C_j \oplus C'_j \in \mathcal{CL}_i$, then we can uniquely determine the 3-bit subkey information related on $(v''_{j1_{up}}, v'_{j2_{up}}, v_{j3_{up}})$ by checking $k_{j1} \oplus c_{L_{j1}} = v''_{j1_{up}}, k_{j2} \oplus c_{L_{j2}} = v'_{j2_{up}}$ and $k_{j3} \oplus c_{L_{j3}} = v_{j3_{up}}$, where $c_{L_{ji}} \in C_L$. We call this procedure uniquely determining the right value for k_{j1}, k_{j2}, k_{j3} '**Determine procedure**' and shortly denote it as ***Determine***(C_j, C'_j). In addition, if we guess $k_{j4}, k_{j5}, k_{j6}, k_{j7}$ and check whether $k_{j4} \oplus c_{L_{j4}} = v''_{j4_{dw}}, k_{j5} \oplus c_{L_{j5}} = v'_{j5_{dw}}, k_{j6} \oplus c_{L_{j6}} = v'_{j6_{dw}}$, and $k_{j7} \oplus c_{L_{j7}} = v_{j7_{dw}}$, then there remain 4 candidates of $k_{j4}, k_{j5}, k_{j6}, k_{j7}$ as a right value with a probability of 1/4. We call this procedure finding all values for $k_{j1}, k_{j2}, k_{j3}, k_{j4}$ '**Find procedure**' and shortly denote it as ***Find***(C_j, C'_j). The following notation '$(k_{j4}, k_{j5}, k_{j6}, k_{j7}) \in$ *Find*(C_j, C'_j)' means that $k_{j4}, k_{j5}, k_{j6}, k_{j7}$ is a possible value for C_j, C'_j. Let $x_1, \cdots, x_n$ be a binary string then $Bi(x_1, \cdots, x_n)$ denotes the number $x_1 * 2^0 + \cdots + x_n * 2^{n-1}$.

Assumption : The attacker knows that $K \oplus K' = (0, 0, e_{64}, e_{64})$

Input : $(P_1, C_1), (P_1, C'_1), (P_2, C_2), (P_2, C'_2), \cdots, (P_{2^{41}}, C_{2^{41}}), (P_{2^{41}}, C'_{2^{41}})$

Output: 20-bit partial key of K_1 and 27-bit partial key information of $K_1 \oplus K_2$ or $K_1 \oplus K_3$

$\cdot\, \mathcal{CL}_1, \mathcal{CL}_2, \cdots, \mathcal{CL}_8$: empty set, $k_1[j] = k_2[j] = k_3[j] = 0$
and $K[Bi(k_4[j], k_5[j], k_6[j], k_7)[j]] = 0$

1. For each j $(1 \le j \le 2^{41})$
   If $C_j \oplus C'_j = (0, e_k)$, for some k
   If k = 3 or 4 or 7 or 8, then $\mathcal{CL}_1 = \mathcal{CL}_1 \cup \{(C_j, C'_j)\}$
   If k = 11 or 12 or 15 or 16, then $\mathcal{CL}_2 = \mathcal{CL}_2 \cup \{(C_j, C'_j)\}$
   If k = 19 or 20 or 23 or 24, then $\mathcal{CL}_3 = \mathcal{CL}_3 \cup \{(C_j, C'_j)\}$
   If k = 27 or 28 or 31 or 32, then $\mathcal{CL}_4 = \mathcal{CL}_4 \cup \{(C_j, C'_j)\}$
   If k = 35 or 36 or 39 or 40, then $\mathcal{CL}_5 = \mathcal{CL}_5 \cup \{(C_j, C'_j)\}$
   If k = 43 or 44 or 47 or 48, then $\mathcal{CL}_6 = \mathcal{CL}_6 \cup \{(C_j, C'_j)\}$
   If k = 51 or 52 or 55 or 56, then $\mathcal{CL}_7 = \mathcal{CL}_7 \cup \{(C_j, C'_j)\}$
   If k = 59 or 60 or 63 or 64, then $\mathcal{CL}_8 = \mathcal{CL}_8 \cup \{(C_j, C'_j)\}$
2. For each $(C_j, C'_j) \in \mathcal{CL}_i$
   2.1 Do $Determine(C_j, C'_j)$
   Output $k_1[j], k_2[j], k_3[j]$
   2.2 Do $Find(C_j, C'_j)$
   2.2.1 For all $(k_4[j], k_5[j], k_6[j], k_7)[j]]) \in Find(C_j, C'_j)$
   $K[Bi(k_4[j], k_5[j], k_6[j], k_7)[j]]] += 1$
   2.2.2 If a $K[Bi(k_4[j], k_5[j], k_6[j], k_7)[j]])] \ge 3$, output $k_4[j], k_5[j], k_6[j], k_7)[j]]$

Algorithm 1 : Related-key differential attack on full-round CIKS-128

4.3 Related-Key Differential Attack on Full-Round CIKS-128H

In this section, we present a related key differential attack on full-round CIKS-128H. First, we can construct a 1-round related key differential characteristic under the same condition represented in Section 4.1. Both probabilities of *C1* and *C2* are replaced 2^{-3} as 2^{-5} by *Property* 11 and 4. Therefore, if we consider the respective encryptions of the plaintext $P = (P_L, P_R)$ under the master keys $\mathsf{K} = (K_1, K_2, K_3, K_4)$ and $\mathsf{K'} = (K_1, K_2, K_3 \oplus e_{64}, K_4 \oplus e_{64})$ and get the corresponding ciphertexts $C = (C_L, C_R)$ and $C' = (C'_L, C'_R)$, then we can construct a full-round related key differential characteristic with a probability 2^{-40} as the following Table 6.

Table 6. A full-round related-key differential characteristic of CIKS-128H

Round	1	2	3	4	5	6	7	8
Case	*C2*	*C1*	*C1*	*C2*	*C2*	*C1*	*C1*	*C2'*

The attack condition of the final round in above table, *C2'*, is equal to that on CIKS-128 mentioned in Section 4.2. That is, the input difference of the $R^{-1}_{64/192}$-box is e_{64} in the final round. Thus, if we get the ciphertext pairs C and C' satisfying $C \oplus C' = e_i$ or $e_{i,i+1}$, then we can obtain the subkey information (Refer to *Property* 7, 9 and 14). The attack scenario of CIKS-128H is almost equal to the method represented in Section 4.2. So, we omit a detail description of the attack procedure.

5 Conclusion

In this paper, we have presented the related-key differential attacks on DDO-based ciphers, CIKS-128 and CIKS-128H which are improvements of CIKS-1. In particular, we have derived many properties of DDO-boxes which are used in our attacks. These works suggests that the greatest possible care has to be taken when proposing improvements of the existing block ciphers. Furthermore, Our works means that when data-dependent operations are very interesting cryptographic primitive, there needs a considerable caution to design a DDO-based block cipher.

References

1. E. Biham and A. Shamir, "Differential Cryptanalysis of the Data Encryption Standard", Springer-Verlag, 1993.
2. N. D. Goots, B. V. Izotov, A. A. Moldovyan, and N. A. Moldovyan, "Modern cryptography: Protect Your Data with Fast Block Ciphers", Wayne, A-LIST Publish., 2003.
3. N. D. Goots, B. V. Izotov, A. A. Moldovyan, and N. A. Moldovyan, "Fast Ciphers for Cheap Hardware : Differential Analysis of SPECTR-H64", *MMM-ACNS'03*, volume 2776 of *Lecture Notes of Computer Science*, Springer-Verlag, 2003, pp. 449-452.
4. N. D. Goots, A. A. Moldovyan, N. A. Moldovyan, "Fast Encryption ALgorithm Spectr-H64", *MMM-ACNS'01*, volume 2052 of *Lecture Notes of Computer Science*, Springer-Verlag, 2001, pp. 275-286.
5. J. Kelsey, B. Schneier, and D. Wagner, "Key Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES", *Advances in Cryptology - CRYPTO '96*, volume 1109 of *Lecture Notes of Computer Science*, Springer-Verlag, 1996, pp. 237-251.
6. J. Kelsey, B. Schneier, and D. Wagner, "Related-Key Cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA", *Proceedings of International Conference on Information and Communications Security (ICICS '97)*, volume 1334 of *Lecture Notes of Computer Science*, Springer-Verlag, 1997, pp. 233-246.
7. J. Kim, G. Kim, S. Hong, S. Lee and D. Hong, *The Related-Key Rectangle Attack - Application to SHACAL-1*, ACISP 2004, LNCS 3108, pp. 123-136, Springer-Verlag, 2004
8. J. Kim, G. Kim, S. Lee, J. Lim and J. Song, *Related-Key Attacks on Reduced Rounds of SHACAL-2*, INDOCRYPT 2004, To appear.

9. Y. Ko, D. Hong, S. Hong, S. Lee, and J. Lim, "Linear Cryptanalysis on SPECTR-H64 with Higher Order Differential Property", *MMM-ACNS03*, volume 2776 of *Lecture Notes of Computer Science*, Springer-Verlag, 2003, pp. 298-307.
10. Y. Ko, S. Hong, W. Lee, S. Lee, and S. Kang, "Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round GOST", *FSE 2004*, volume 3017 of *Lecture Notes of Computer Science*, Springer-Verlag, 2004, pp. 299-316.
11. Y. Ko, C. Lee, S. Hong and S. Lee, "Related Key Differential Cryptanalysis of Full-Round SPECTR-H64 and CIKS-1 ", *ACISP 2004*, volume 3108 of *Lecture Notes of Computer Science*, Springer-Verlag, 2004, pp. 137-148.
12. C. Lee, D. Hong, S. Lee, S. Lee, H. Yang, and J. Lim, "A Chosen Plaintext Linear Attack on Block Cipher CIKS-1", *The 4th International Conference, ICICS 2002*, volume 2513 of *Lecture Notes of Computer Science*, Springer-Verlag, 2002, pp. 456-468.
13. M. Matsui, "Linear cryptanalysis method for DES cipher", *Advances in Cryptology - EUROCRYPTO'93*, volume 765 of *Lecture Notes of Computer Science*, Springer-Verlag, 1993, pp. 386-397.
14. A. A. Moldovyan and N. A. Moldovyan, "A cipher Based on Data-Dependent Permutations", *Journal of Cryptology*, volume 15, no. 1 (2002), pp. 61-72
15. N. Sklavos, A. A. Moldovyan, and O. Koufopavlou, "Encryption and Data Dependent Permutations : Implementation Cost and Performance Evaluation", *MMM-ACNS'03*, volume 2776 of *Lecture Notes of Computer Science*, Springer-Verlag, 2003, pp. 337-348.
16. N. Sklavos, N. A. Moldovyan, and O. Koufopavlou, "A New DDP-based Cipher CIKS-128H: Architecture, Design & VLSI Implementation Optimization of CBC-Encryption & Hashing over 1 GBPS", proceedings of *The 46th IEEE Midwest Symposium on Circuits & Systems*, December 27-30, Cairo, Egypt, 2003.
17. N. Sklavos and O. Koufopavlou, "Data Dependent Rotations, a Trustworthy Approach for Future Encryption Systems/Ciphers: Low Cost and High Performance", *Computers and Security, Elsevier Science Journal*, Vol.22, No 7, 2003.

Cryptanalysis of Ake98

Jorge Nakahara Júnior[1] and Daniel Santana de Freitas[2]

[1] jorge_nakahara@yahoo.com.br
[2] LabSEC, Laboratório de Segurança em Computação, UFSC, Brazil
santana@inf.ufsc.br

Abstract. This paper describes a linear attack on the Ake98 block cipher, an updated version of the Akelarre cipher presented by Alvarez *et al.* at the SAC'96 Workshop. The new attacks require the assumption of weak keys. It is demonstrated that Ake98 does not introduce enough security measures to counter cryptanalytic attacks, both in a known-plaintext and in a ciphertext-only setting. A key-recovery attack on 4.5-round Ake98, for instance, is applicable to a weak-key class of size 2^{108}, and requires only 71 known plaintexts, with an effort of $71 \cdot 2^{71}$ half-round decryptions. Moreover, the existence of weak keys precludes the use of Ake98 as a building block for other cryptographic primitives, such as in Davies-Meyer Hash mode. Attacks using weak keys can be applied up to 11.5 rounds of Ake98 with less effort than an exhaustive key search. But, Ake98 with 8.5 rounds is already slower than IDEA, RC6 or AES, which implies that this updated version of the Akelarre cipher does not seem to provide significant advantages (security or efficiency) compared to the former, more established ciphers.

Keywords: cryptanalysis, Akelarre, Ake98, IDEA, RC5, RC6, AES.

1 Introduction

Akelarre is a block cipher designed by Alvarez *et al.* [4] and presented at SAC'96 Workshop. Akelarre combines design features from the IDEA [9] and RC5 [11] ciphers, and processes 128-bit text blocks, uses a 128-bit key, and iterates 4 rounds plus an output transformation (OT). The operations of modular addition, $\boxplus$, and exclusive-or, $\oplus$, were inherited from IDEA, while bitwise rotation, $\lll$, came from RC5. In [8], Knudsen and Rijmen presented known-plaintext and ciphertext-only attacks on Akelarre for any number of rounds, and that are independent of the key schedule algorithm. Further attacks were also presented by Ferguson and Schneier in [6], but using chosen plaintext.

Subsequently, the designers of Akelarre presented Ake98 [3] that is claimed to avoid the previous attacks on Akelarre.

This paper is organized as follows: Sect. 2 describes briefly the Akelarre block cipher; Sect. 3 describes Ake98 and the main differences with Akelarre. Sect. 4 explains the attack on Ake98, its similarity to the attack of Knudsen-Rijmen, the attack requirements and its complexity. Subsect. 4.2 describes a ciphertext-only attack on Ake98. Sect. 5 compares the software performance of Ake98 with that of AES, IDEA and RC6. Sect. 6 concludes the paper.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 206–217, 2004.

2 The Akelarre Cipher

The Akelarre block cipher was presented at the SAC'96 workshop, and its design combines features from the IDEA and RC5 ciphers. Akelarre uses three operations on w-bit words: bitwise exclusive-or, denoted $\oplus$, addition modulo 2^w, denoted $\boxplus$, and bitwise rotation, denoted $\lll$. The multiplication operation of IDEA is absent. A note on terminology: the notation $\mathrm{lsb}_i(X)$ (lower case) will denote the i-th least significant bit(s) of X, while $\mathrm{LSB}_j(X)$ (upper case) will denote the ensemble of i consecutive least significant bits[1].

All of the internal operations in Akelarre are on w-bit words. Akelarre operates on variable-length words, text blocks and keys, and uses a variable number of rounds. The suggested parameter values in [4] are: 128-bit blocks, 32-bit words, 128-bit key and 4 rounds. Fig. 1 depicts the computational graph of Akelarre. The MA-box of IDEA becomes an AR-box (Addition-Rotation box). Details of the AR-box are given in the Appendix.

The key schedule algorithm of Akelarre will not be described in this paper but the interested reader can find further information in [4].

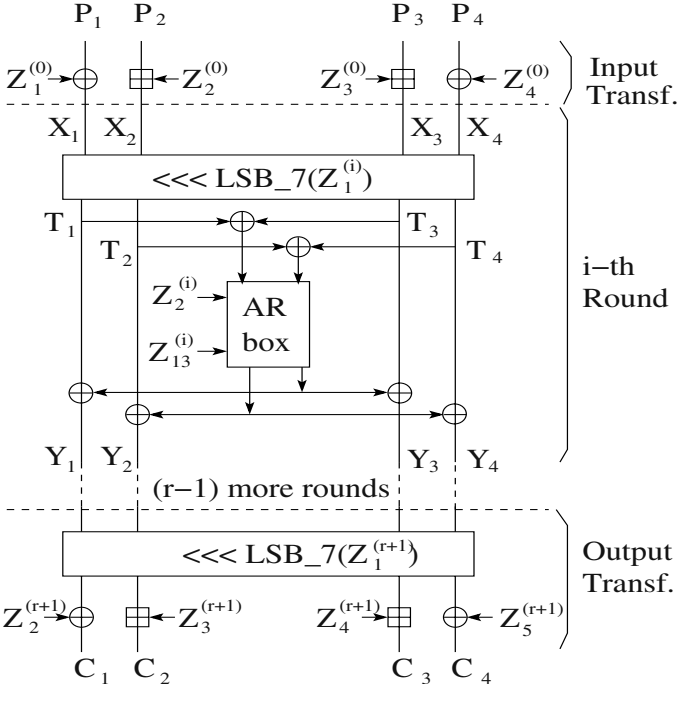

Fig. 1. Computational graph of the Akelarre block cipher

[1] For example, if $X = 01101110_2$ in binary, then $\mathrm{lsb}_1(X) = 0$, $\mathrm{lsb}_2(X) = 1$, but $\mathrm{LSB}_2(X) = 2$, and $\mathrm{LSB}_3(X) = 6$.

3 The Ake98 Cipher

In [3], an updated version of Akelarre, called Ake98, was presented. It is claimed that Ake98 resists the attacks made formerly on Akelarre [6, 8]. Ake98 differs from Akelarre in the new AR-box (Addition-Rotation box), in the swapping of words at the end of a round, and the addition of subkeys in the beginning of each round. Fig. 2 depicts the computational graph of Ake98. Details of the AR-box of Ake98 are provided in the Appendix (Sect. 5).

The block and key sizes, the number of rounds, and the internal word sizes in Ake98 are variable but no minimum value is set by the authors for any parameter. For comparison purposes, the same parameter values for Akelarre will also be assumed for Ake98.

The key schedule of Ake98 will not be described here. The only property assumed for the key schedule of Ake98 is that it behaves as a pseudo-random number generator. Further details of the subkey generation in Ake98 can be found in [3].

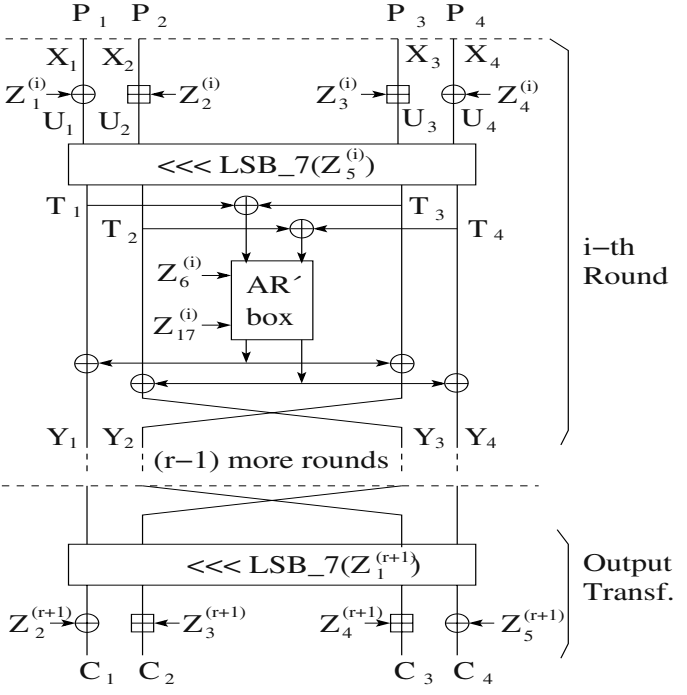

Fig. 2. Computational graph of the Ake98 cipher

4 A Known-Plaintext Attack on Ake98

The Knudsen-Rijmen attack [8] on Akelarre exploited the fact that the leftmost input to the AR-box can be computed from just two input and output words in

a round. From Fig. 1, $T_1 \oplus T_3 = Y_1 \oplus Y_3$ for the i-th round. Similarly, $T_2 \oplus T_4 = Y_2 \oplus Y_4$. These relations can be extended across the key-dependent rotation as[2]

$$(Y_1 \oplus Y_3)|(Y_2 \oplus Y_4) = ((X_1 \oplus X_3)|(X_2 \oplus X_4)) \lll Z_1^{(i)} . \tag{1}$$

Relation (1) always holds, independent of the round subkeys and of the AR-box. Moreover, this is an iterative relation, namely it can be combined with itself. The attack of [8] uses (1) as an invariant for the full Akelarre, except for the input and output transformations (Fig. 1). Notice that this attack applies to any number of rounds.

Notice that (1) does not hold for the IDEA and PES [9] ciphers because of the addition and the multiplication operations (Akelarre and Ake98 do not use multiplication).

For Ake98 similar relations to (1) can be obtained, under weak subkey assumptions. Observe that in Fig. 2, $T_1 \oplus T_3 = Y_1 \oplus Y_2$, and $T_2 \oplus T_4 = Y_3 \oplus Y_4$. To achieve a similar relation to (1), both of them are combined, resulting in

$$T_1 \oplus T_2 \oplus T_3 \oplus T_4 = Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 . \tag{2}$$

Furthermore, across the key-dependent rotation:

$$Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4 = (U_1 \oplus U_2 \oplus U_3 \oplus U_4) \lll Z_5^{(i)} , \tag{3}$$

where it is implicitly assumed that only the least significant seven bits of $Z_5^{(i)}$ are used as the rotation amount.

Relation (3) does not hold in general across the modular addition with subkeys at the beginning of a round, but it still holds with certainty for the least significant bit, because of the absence of a carry bit[3]:

$$\mathrm{lsb}_1(Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4) = \mathrm{lsb}_{-Z_5^{(i)} \bmod 32+1}(X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus Z_1^{(i)} \oplus Z_2^{(i)} \oplus Z_3^{(i)} \oplus Z_4^{(i)}) . \tag{4}$$

Relation (4) is not iterative, but under the assumption that $\mathrm{LSB}_7(Z_5^{(i)}) \in \{0, 32, 64, 96\}$, that is, a rotation amount that is a multiple of the word size of Ake98, this relation can be rewritten as:

$$\mathrm{lsb}_1(Y_1 \oplus Y_2 \oplus Y_3 \oplus Y_4) = \mathrm{lsb}_1(X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus Z_1^{(i)} \oplus Z_2^{(i)} \oplus Z_3^{(i)} \oplus Z_3^{(i)}) , \tag{5}$$

which is iterative, and independent of the new AR-box. Iterating relation (5) four times, results in a probabilistic distinguisher, under the weak subkey assumptions: $\mathrm{LSB}_7(Z_5^{(1)}), \mathrm{LSB}_7(Z_5^{(2)}), \mathrm{LSB}_7(Z_5^{(3)}), \mathrm{LSB}_7(Z_5^{(4)}) \in \{0, 32, 64, 96\}$. Assuming that the key schedule algorithm of Ake98 can be modeled as a pseudo-random number generator, each of the weak subkey assumptions will be taken independently. Therefore, the probability that these assumptions hold for four consecutive rounds is approximated as $(4/2^7)^4 = 2^{-20}$ since there are 2^7 possible

[2] The vertical bar '|' stands for concatenation.

[3] Parameters of the 'lsb' function are counted from 1 up to 32.

rotation amounts. Similarly, under the assumption of a random behavior of the key schedule of Ake98, the weak subkey assumptions are expected to hold for a class of $2^{128} \cdot 2^{-20} = 2^{108}$ (weak) user keys.

For 4-round Ake98, a 1-bit invariant, using relation (5), can be constructed:

$$\begin{aligned} &\mathrm{lsb}_1(P_1 \oplus P_2 \oplus P_3 \oplus P_4) \oplus \mathrm{lsb}_1(C_1 \oplus C_2 \oplus C_3 \oplus C_4) = \\ &\mathrm{lsb}_1(Z_1^{(1)} \oplus Z_2^{(1)} \oplus Z_3^{(1)} \oplus Z_4^{(1)}) \oplus \mathrm{lsb}_1(Z_1^{(2)} \oplus Z_2^{(2)} \oplus Z_3^{(2)} \oplus Z_4^{(2)}) \oplus \\ &\mathrm{lsb}_1(Z_1^{(3)} \oplus Z_2^{(3)} \oplus Z_3^{(3)} \oplus Z_4^{(3)}) \oplus \mathrm{lsb}_1(Z_1^{(4)} \oplus Z_2^{(4)} \oplus Z_3^{(4)} \oplus Z_4^{(4)})\,. \end{aligned} \tag{6}$$

Notice in (6) that for a fixed key, one bit of information on the key, namely, $\mathrm{lsb}_1(\oplus_{i,j=1}^{4} Z_i^{(j)})$, can be recovered given one bit $\mathrm{lsb}_1(C_1 \oplus C_2 \oplus C_3 \oplus C_4)$ of ciphertext information and of the plaintext, $\mathrm{lsb}_1(P_1 \oplus P_2 \oplus P_3 \oplus P_4)$; or alternatively, given the plaintext, and an unknown key, one bit of information on the ciphertext can be obtained with certainty.

Relation (6) alone can be used to distinguish 4-round Ake98 (under weak key assumptions and without the OT) from a random permutation, using only known plaintext/ciphertext pairs.

Moreover, (6) can be used in a key-recovery attack, to discover the subkeys of the OT. If we call the output transformation a half-round, this is a 0.5R attack. The corresponding 1-bit distinguisher is:

$$\begin{aligned} &\mathrm{lsb}_1(P_1 \oplus P_2 \oplus P_3 \oplus P_4) \oplus \mathrm{lsb}_1(Z_1^{(1)} \oplus Z_2^{(1)} \oplus Z_3^{(1)} \oplus Z_4^{(1)}) \oplus \\ &\mathrm{lsb}_1(Z_1^{(2)} \oplus Z_2^{(2)} \oplus Z_3^{(2)} \oplus Z_4^{(2)}) \oplus \mathrm{lsb}_1(Z_1^{(3)} \oplus Z_2^{(3)} \oplus Z_3^{(3)} \oplus Z_4^{(3)}) \oplus \\ &\mathrm{lsb}_1(Z_1^{(4)} \oplus Z_2^{(4)} \oplus Z_3^{(4)} \oplus Z_4^{(4)}) = \\ &\mathrm{lsb}_1((C_1 \oplus Z_2^{(5)} \oplus C_2 \boxminus Z_3^{(5)} \oplus (C_3 \boxminus Z_4^{(5)}) \oplus (C_4 \oplus Z_5^{(5)})) \ggg \mathrm{LSB}_5(Z_1^{(5)}))\,. \end{aligned} \tag{7}$$

In a known-plaintext setting, the unknowns in (7) are[4] $\mathrm{LSB}_5(Z_1^{(5)})$, $Z_2^{(5)} \oplus Z_5^{(5)}$, $Z_4^{(5)}$, $Z_3^{(5)}$ and $\mathrm{lsb}_1(\oplus_{i,j=1}^{4} Z_i^{(j)})$. Actually, only the $\mathrm{LSB}_5(Z_1^{(5)})$-th bit of $Z_2^{(5)} \oplus Z_5^{(5)}$ is required. In total, $5+1+32+32 = 70$ subkey bits can be recovered, and the effort for each of the 2^{70} subkey candidates is equivalent to decrypting the OT, or a half-round computation.

The amount of known plaintext (KP) needed for the attack is computed as follows. Once the 70 subkey bits are guessed correctly in (7), the combined value of plaintext, ciphertext and guessed subkey bits must match the 1-bit key-dependent invariant:

$$\mathrm{lsb}_1(\oplus_{i,j=1}^{4} Z_i^{(j)})\,. \tag{8}$$

The value of (8) is unknown, but is constant for a fixed key. Therefore, the correct 70 subkey bits must always give a constant value, whatever the plaintext, while the wrong 70 subkey bits will only match (8) with a probability of 1/2. This reasoning is based on the fact that the correct subkey value actually decrypts

[4] Even though the least significant 7 bits of $Z_1^{(5)}$ are used in a block, the invariant involves (the xor of) 32-bit words, thus only the 5 least significant bits of $Z_1^{(5)}$ are relevant.

the OT, reducing the 4.5 rounds to 4 rounds, where the distinguisher can be checked; but, the wrong subkey will not decrypt the OT correctly, rather, it will add a further 0.5 rounds on top of the 4.5-round Ake98, and its 1-bit result shall be (more) random. Therefore, the expected number of false alarms (subkeys) surviving this filtering after 71 known plaintext/ciphertext pairs are used, is $2^{70} \cdot (\frac{1}{2})^{71} < 1$.

The attack using the 4-round distinguisher (7) was applied to a 4.5-round Ake98 and not to 5.5 rounds, because the latter would require too many subkey bits to recover simultaneously, namely the subkeys of one round plus the OT. The distinguisher for 5.5-round Ake98 would be:

$$\begin{aligned}
&\mathrm{lsb}_1(P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus Z_1^{(1)} \oplus Z_2^{(1)} \oplus Z_3^{(1)} \oplus Z_4^{(1)}) \oplus \\
&\mathrm{lsb}_1(Z_1^{(2)} \oplus Z_2^{(2)} \oplus Z_3^{(2)} \oplus Z_4^{(2)}) \oplus \mathrm{lsb}_1(Z_1^{(3)} \oplus Z_2^{(3)} \oplus Z_3^{(3)} \oplus Z_4^{(3)}) \oplus \\
&\mathrm{lsb}_1(Z_1^{(4)} \oplus Z_2^{(4)} \oplus Z_3^{(4)} \oplus Z_4^{(4)}) = \\
&\mathrm{lsb}_1((((C_1 \oplus Z_2^{(6)}) \ggg \mathrm{LSB}_5(Z_1^{(6)}) \oplus Z_1^{(5)}) \oplus \\
&((C_2 \boxminus Z_3^{(6)}) \ggg \mathrm{LSB}_5(Z_1^{(6)}) \boxminus Z_2^{(5)}) \oplus \\
&((C_3 \boxminus Z_4^{(6)}) \ggg \mathrm{LSB}_5(Z_1^{(6)}) \boxminus Z_3^{(5)}) \oplus \\
&((C_4 \oplus Z_5^{(6)}) \ggg \mathrm{LSB}_5(Z_1^{(6)}) \oplus Z_4^{(5)})) \ggg \mathrm{LSB}_5(Z_5^{(5)}))\,.
\end{aligned} \tag{9}$$

Note that a 1.5R-attack on 5.5-round Ake98 using (9) would require guessing one bit of $Z_1^{(5)} \oplus Z_4^{(5)}$, $Z_2^{(6)}$, and $Z_5^{(6)}$; the full 32 bits of $Z_2^{(5)}$, $Z_3^{(5)}$, $Z_3^{(6)}$, $Z_4^{(6)}$, and $\mathrm{LSB}_5(Z_1^{(6)})$, $\mathrm{LSB}_5(Z_5^{(5)})$, or 141 subkey bits simultaneously.

In general, the more rounds are attacked the smaller the weak key class. Table 1 lists the number of weak keys for attacks on a different number of rounds of Ake98. All the attacks require about 71 known plaintext/ciphertext pairs, and effort equivalent to $71 \cdot 2^{71}$ decryptions of the OT. Assuming the OT is about half a round, it represents $\frac{1}{9}$ of a 4.5-round encryption. Thus, the attack complexity is equivalent to $\frac{1}{9} \cdot 71 \cdot 2^{71} \approx 8 \cdot 2^{71} = 2^{74}$ 4.5-round encryptions.

Table 1. Estimated effort and weak key class size, $|WKC|$, in attacks on Ake98

# Rounds	$\lvert WKC\rvert$	Attack Effort
4.5	2^{108}	2^{74}
6.5	2^{98}	$\frac{1}{13} \cdot 71 \cdot 2^{71} \approx 2^{73}$
8.5	2^{88}	$\frac{1}{17} \cdot 71 \cdot 2^{71} \approx 2^{73}$
10.5	2^{78}	$\frac{1}{21} \cdot 71 \cdot 2^{71} \approx 2^{73}$
11.5	2^{73}	$\frac{1}{23} \cdot 71 \cdot 2^{71} \approx 2^{72.5}$
12.5	2^{68}	$\frac{1}{25} \cdot 71 \cdot 2^{71} \approx 2^{72.5}$
25.5	2^{3}	$\frac{1}{51} \cdot 71 \cdot 2^{71} \approx 2^{71}$

From Table 1, in order to avoid attacks based on weak keys, Ake98 should have more than 25.5 rounds. Nonetheless, our attack is more efficient than exhaustive key search (in a weak-key class) up to 11.5 rounds. For 12.5-round

Ake98, the exhaustive key search effort for a weak-key class of size 2^{68} is less than the $2^{72.5}$ encryptions of our attack.

As the last remark, the attack described in this section, although explained for a 32-bit word version of Ake98, applies similarly to other word sizes.

4.1 New Weak Key Classes

The rationale for choosing subkeys that cause rotations by multiples of 32 bits can be further extended to weak subkeys whose values are of the form $16 + 32t$, $0 \leq t \leq 3$. In this case, a one-round relation with input (X_1, X_2, X_3, X_4) and output $(Y_1^{(1)}, Y_2^{(1)}, Y_3^{(1)}, Y_4^{(1)})$ becomes:

$$\mathrm{lsb}_1(X_1 \oplus X_2 \oplus X_3 \oplus X_4) = \mathrm{lsb}_1((Y_1^{(1)} \oplus Y_2^{(1)} \oplus Y_3^{(1)} \oplus Y_4^{(1)}) \ggg 16)\,, \quad (10)$$

which is not iterative. But, if two consecutive rounds have block rotations by amounts of the form $16 + 32t$, $0 \leq t \leq 3$, then for the next round:

$$\mathrm{lsb}_1(Y_1^{(1)} \oplus Y_2^{(1)} \oplus Y_3^{(1)} \oplus Y_4^{(1)}) = \mathrm{lsb}_1((Y_1^{(2)} \oplus Y_2^{(2)} \oplus Y_3^{(2)} \oplus Y_4^{(2)}) \ggg 16)\,, \quad (11)$$

where $Y_i^{(2)}$, $1 \leq i \leq 4$ are the output words after two rounds. Combining (10) and (11) results in:

$$\mathrm{lsb}_1(X_1 \oplus X_2 \oplus X_3 \oplus X_4) = \mathrm{lsb}_1(Y_1^{(2)} \oplus Y_2^{(2)} \oplus Y_3^{(2)} \oplus Y_4^{(2)})\,, \quad (12)$$

which has the following properties:

- it is a 2-round iterative linear relation, in contrast to (5) which is 1-round iterative;
- it holds with a probability that depends on the carry bit of addition with the two subkeys in the middle of a block between rounds, $Z_2^{(i)}$ and $Z_3^{(i)}$. The probability that there is no carry from the 15-th to the 16-th bit is $p = 1/2 + 1/2^{17}$. Assuming the subkey values are independent, the probability of (12) holding is approximated as $p^2 \approx 2^{-2}$.

Thus, this new weak-subkey assumption implies another, new weak-key class, namely the one which generates rotations of the form $16 + 32t$, $0 \leq t \leq 3$, distinct from the original weak-key class that generates rotations of the form $32t$, $0 \leq t \leq 3$. The former weak-key class, though, is less effective since it holds with a lower probability, 2^{-2} every two rounds, than the latter.

It is straightforward to deduce other rotation amounts, for example, $8 + 32t$, $0 \leq t \leq 3$, which lead to further new weak-key classes, with exponentially lower probability compared to (5) and (12) due to the carry bits of addition with subkeys. These linear relations become iterative for 4, 8 or more rounds.

The rotation amounts mentioned previously do not need to be powers of 2 plus a multiple of 32. More generally, the rotation amounts can be of the form $i + 32t$, $0 \leq t \leq 3$, $0 \leq i \leq 31$. The main point for deducing any of these linear relations is to track the exact position of the least significant bit of each 32-bit word in a block, because, once this bit is correctly located, the xor of 32-bit words in (5) can be applied.

Consequently, the key space can be split into several sets of disjoint weak-key classes, one for each possible set of rotation amounts, and several of them correspond to keys which are susceptible to an attack similar to that in Sect. 4.

4.2 Ciphertext-Only Attack on Ake98

The attack on 4.5-round Ake98 presented in Sect. 4 can also be adapted to recover subkeys at the top (plaintext) end of Ake98.

The fact that only a few bits of the plaintext blocks are needed for the attack motivates a ciphertext-only (CO) approach to attack Ake98. We assume that ciphertext is always known by any adversary. Further, assume that the plaintext is known to be ASCII text, and some probable phrases (16 bytes long, that is, the block size of Ake98) are suspected to occur regularly in the plaintext, for instance, "replyimmediately" or "tocommandergeneral".

The ciphertext-only attack on 4.5-round Ake98 assumes that the last four block rotations, including the one in the OT are a multiple of the word size (32 bits), instead of the first four block rotations. The distinguisher is similar to (7):

$$\begin{aligned} &\mathrm{lsb}_1(C_1 \oplus C_2 \oplus C_3 \oplus C_4) \oplus \mathrm{lsb}_1(Z_1^{(2)} \oplus Z_2^{(2)} \oplus Z_3^{(2)} \oplus Z_4^{(2)}) \oplus \\ &\mathrm{lsb}_1(Z_1^{(3)} \oplus Z_2^{(3)} \oplus Z_3^{(3)} \oplus Z_4^{(3)}) \oplus \mathrm{lsb}_1(Z_1^{(4)} \oplus Z_2^{(4)} \oplus Z_3^{(4)} \oplus Z_4^{(4)}) \oplus \\ &\mathrm{lsb}_1(Z_1^{(5)} \oplus Z_2^{(5)} \oplus Z_3^{(5)} \oplus Z_4^{(5)}) = \\ &\mathrm{lsb}_1((P_1 \oplus Z_1^{(1)} \oplus P_2 \oplus Z_2^{(1)} \oplus (P_3 \boxplus Z_3^{(1)}) \oplus (P_4 \boxplus Z_4^{(1)})) \ggg \mathrm{LSB}_5(Z_5^{(1)})) \,. \end{aligned} \tag{13}$$

Thus, (13) only requires the xor of some least significant bits of the plaintext blocks, namely, only some small statistical information. The time complexity and amount of probable texts for this attack are the same as for the attack in Sect. 4, requiring about 71 (probable) 16-byte long plaintexts encrypted under a fixed key. Similar attacks apply to more rounds of Ake98, under the appropriate weak subkey assumptions.

5 Software Performance of Ake98

Table 2 lists the main parameters and the performance in software of Ake98, AES, IDEA and RC6 block ciphers, for comparison. Performance estimates for encryption and key schedule were measured in CPU cycles per byte encrypted on an AMD Duron 1.2 GHz, 512 MB RAM and 128 MB cache memory, under Linux, and using the gcc compiler ver. 3.2.2 with optimization option -O3. Measurements were obtained from 2^{16} up to 2^{26} blocks encrypted under each cipher.

From Table 2, the performance figures indicate that 4.5-round Ake98 is faster than 8.5-round IDEA, but slower than 10-round AES and 20-round RC6 (standard parameters), under the same test conditions.

Moreover, Table 3 shows that the software performance of Ake98 for increasing number of rounds degrades sharply. For 8 rounds, Ake98 is not faster than any of the three previously mentioned ciphers. For more than 25.5 rounds, Ake98 is not expected to have any weak key, but then it becomes about four times slower than

Table 2. Software performance and main parameters of some block ciphers

Cipher	Ake98	AES	IDEA	RC6-w/r/b
Operations	$\oplus, \boxplus, \lll$	$\oplus$,xtime,S-box	$\oplus, \boxplus, \odot$	$\oplus$, *, $\lll$
Block Size (bits)	variable	128	64	$4w$
Key Size (bits)	$64t$	128, 192, 256	128	$8b$
#Rounds	variable♯	10, 12, 14	8.5	r, $r = 20$ (AES)
Origin	Alvarez *et al.*	Daemen, Rijmen	Lai, Massey, Murphy	Rivest *et al.*
Year	2000	1998	1991	1998
Word Size (bits)	variable	8	16	w, $w = 32$ (AES)
Cipher Structure	IDEA+RC5	SPN	own	Feistel
Key Schedule Oper.	$\boxplus$, modular squaring	$\oplus$, S-box	bit permutation	byte permutation
Reference	[3]	[5]	[9]	[12]
Encryption Speed	73	55	93	30

♯: 4.5 rounds, 128-bit block, 128-bit key.

Table 3. Software performance of variable-round Ake98

# Rounds Ake98	4.5	8.5	12.5	16.5	20.5	24.5	28.5
# CPU cycles/byte	73	142	212	283	354	427	499

IDEA, eight times slower than AES, and more than 14 times slower than RC6. Moreover, with more than 427 cycles/byte, the performance of Ake98 becames worse than that of all NESSIE block cipher candidates, except GrandCru [1–p. 53–55].

6 Conclusions

This report presented the first[5] known-plaintext and ciphertext-only attacks on Ake98. In a key-recovery attack, the subkeys of the OT can be recovered with only 71 known plaintext/ciphertext pairs. The attacks are independent of the redesigned AR-box, and can be applied up to 11.5 rounds with less effort than an exhaustive key search. To avoid weak keys, Ake98 would need more than 25.5 rounds, but then its performance degrades sharply.

The attacks in Sect. 4 exploited two main weaknesses of Ake98: the key schedule algorithm did not make any provision to avoid the key-dependent rotation amounts to be multiples of 32 (the word size), even for consecutive rounds; moreover, the subkey mixing operations at the beginning of a round allows invariants involving only the least significant text bits, similar to the attack of [8]. These attacks perhaps could be avoided, for example, if the key schedule algorithm had guaranteed that the rotation amounts were both text and key dependent, such as in RC6 [12].

[5] The authors are not aware of any other attack on Ake98, under any assumption.

Another important observation is that even if the rotation amounts were properly generated, the encryption and decryption structures of Ake98 would still not be reciprocal, that is, the computational graphs for encryption and decryption of Ake98 are different, because the modular addition and bit rotation operations do not commute. Thus, the computational graph does not become an involution by simply transforming the subkeys, as in IDEA.

The existence of weak subkeys for Ake98 are far reaching. Even though the class of 2^{108} weak keys represent only a fraction of 2^{-20} of the key space, it implies, for instance, that Ake98 might not be used as a building block of other cryptographic primitives, such as in Davies-Meyer or Matyas-Meyer-Oseas hash function constructions [10–p. 340, Cap. 9] because the key input depends on the input message string or intermediate hash values, and they can be manipulated to cause weak rotations as in the attacks of Sect. 4.

Further the weak-key class size ($|WKC|$) and type of attacks on IDEA and Ake98 are compared in Table 4. Notice that Hawkes' attacks on IDEA [7] require chosen plaintext (CP), Boomerang attacks on IDEA [2] require chosen plaintext adaptively-chosen ciphertext (CPACC), while the attacks on Ake98, in this paper, require known plaintext (KP) or ciphertext only (CO). It can be noticed additionally in Table 4 that the weak-key class sizes for Ake98 are bigger than for IDEA. Therefore, the attacks on Ake98 apply not only to larger weak-key classes but also work under much more realistic assumptions than on IDEA.

Table 4. Comparison of weak-key class sizes for IDEA and Ake98

			# Rounds					
Attack	Cipher	Type	4	4.5	5	5.5	6	8.5
Hawkes	IDEA	CP	2^{99}	2^{97}	2^{84}	2^{82}	2^{82}	2^{63}
Boomerang	IDEA	CPACC	2^{104}	2^{103}	2^{97}	2^{97}	2^{83}	2^{64}
this paper	Ake98	KP/CO	2^{108}	2^{103}	2^{103}	2^{98}	2^{98}	2^{83}

Additionally, if Ake98 were used in (full 128-bit) OFB and CFB modes of operation [10], then the use of weak keys at the beginning of every round would result in the exclusive-or of the LSBs of the four input words to match the xor of the LSBs of the four output words. This invariant would propagate to the ciphertext, actually revealing information on the plaintext. Since Ake98 with only 4.5 rounds is already slower than the AES and RC6, even its practical usefulness for confidentiality purposes becomes jeopardized.

As the last comment, it is not straightforward to determine which 128-bit user key(s) lead to subkeys that cause weak rotations at the beginning of each round, but the key schedule algorithm does not have any provision to avoid such weak subkeys. It is left as an open problem to discover which 128-bit Ake98 key(s) can lead to weak subkeys.

Acknowledgements

Many thanks to Prof. S.W. Song of the CS Dept. of the Institute of Mathematics and Statistics of the University of São Paulo, Brazil, for the kind logistical support for this research, and to the anonymous referees for the many useful comments.

Appendix

This appendix shows the AR-boxes (Addition-Rotation boxes) of Akelarre and Ake98 (Fig. 3). For the left-rotation operation, $\lll$, the rotation amounts are 4- or 5-bit values from parts of P_1 and Q_2. In Fig.3(a), rotations affect 32-bit operands. For example, the first rotation of P_2 to the left is by an amount represented by the five bits $P_1[1 \ldots 5]$. In Fig.3(b), the rotations affect operands 31 bits wide, namely excluding the most or the least significant bits (darkened in the pictures). These pictures are described for illustrative purposes only, because the attacks in this paper are independent of the AR-boxes.

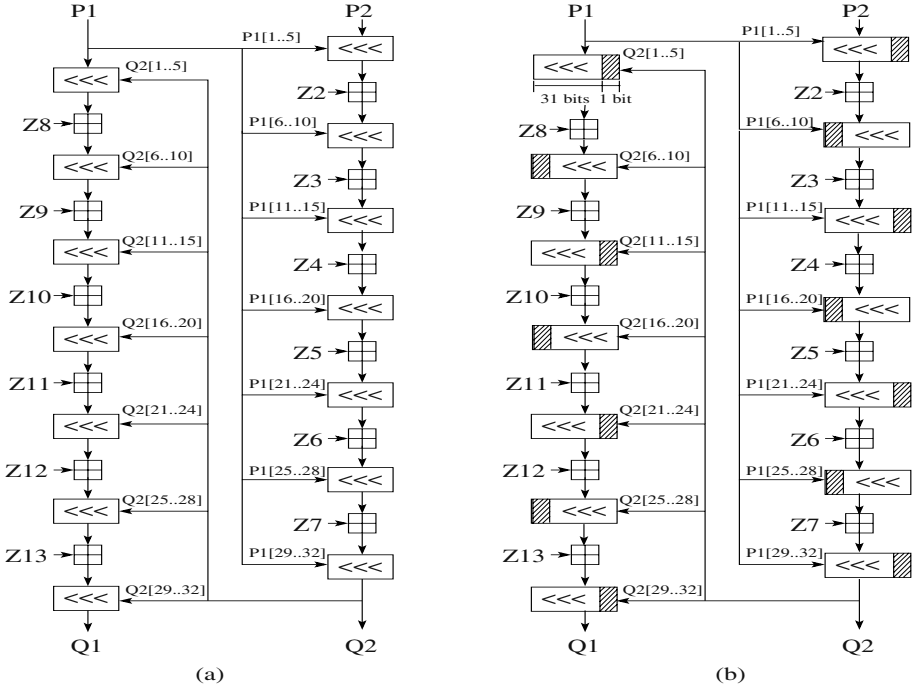

Fig. 3. AR-box of Ake98 (a), and of Akelarre (b)

References

1. E. Biham, "Performance of Optimized Implementations of the NESSIE Primitives," http://cryptonessie.org, Oct. 2002.
2. A. Biryukov, J. Nakahara,Jr., B. Preneel, J. Vandewalle, "New Weak-Key Classes of IDEA," ICICS 2002, R. Deng, S. Qing, F. Bao, J. Zhou, Eds., Springer-Verlag, LNCS 2513, Dec. 2002, 315–326.
3. G. Álvarez Marañón, "Contribución al estudio de la estructura interna del conjunto de Mandelbrot y aplicaciones en criptografía," Facultad de Informática, Universidad Politécnica de Madrid, Sep. 2000, PhD Dissertation, http://www.iec.csic.es/∼gonzalo.
4. G. Alvarez, D. de la Guia, F. Montoya, A. Peinado, "Akelarre: a new Block Cipher Algorithm," 3rd Selected Areas in Cryptography (SAC) Workshop, 1996, 1–14.
5. J. Daemen, V. Rijmen, "AES Proposal: Rijndael," First AES Conference, California, USA, 1998, http://www.nist.gov/aes.
6. N. Ferguson, B. Schneier, "Cryptanalysis of Akelarre," 4th Selected Areas in Cryptography (SAC) Workshop, 1997, 201–212.
7. P.M. Hawkes, "Asymptotic Bounds on Differential Probabilities and an Analysis of the Block Cipher IDEA," PhD Dissertation, The University of Queensland, St. Lucia, Australia, Dec. 1998.
8. L.R. Knudsen and V. Rijmen, "Ciphertext-Only Attack on Akelarre," Cryptologia, vol. XXIV, no. 2, Apr. 2000, 135–147.
9. X. Lai, "On the Design and Security of Block Ciphers," ETH Series in Information Processing, J.L. Massey, Ed., Vol. 1, 1995, Hartung-Gorre Verlag, Konstanz.
10. A.J. Menezes, P.C. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.
11. R.L. Rivest, "The RC5 Encryption Algorithm," B. Preneel, Ed., 2nd Fast Software Encryption Workshop, 1995, Springer-Verlag, LNCS 1008, 86–96.
12. R.L. Rivest, M.J.B. Robshaw, R. Sidney, Y.L. Yin, "The RC6 Block Cipher," First AES Conference, California, USA, 1998, http://csrc.nist.gov/encryption/aes/.

Designing an Efficient and Secure Public-Key Cryptosystem Based on Reducible Rank Codes

Thierry Berger[1] and Pierre Loidreau[2]

[1] LACO, Université de Limoges, France
Thierry.Berger@unilim.fr
[2] Ecole Nationale Supérieure de Techniques Avancées (ENSTA), France
Pierre.Loidreau@ensta.fr

Abstract. In this paper we modify the cryptosystem presented in [16] based on the problem of decoding in rank metric. We design a cryptosystem more secure than the original one with a better transmission rate. We show that this system resists to the *message resend* and *reaction* attacks, and can be used with a small public-key (around 10kbits).

1 Introduction

Cryptosystems based on coding theory were first introduced by McEliece in 1978 [13], just a few months after the RSA cryptosystem was published. This system uses as private-key a Goppa code known to have fast polynomial-time decoding algorithms up to its error-correcting capability, and two scrambling non-singular matrices. The public-key is a generator matrix of the scrambled Goppa code.

Despite quite a number of attempts since 1978, the original system remains unbroken. However it is not widely used. Namely, the main problem of cryptosystems basing their security on the problem of decoding in Hamming metric is that the efficiency of general decoding algorithms implies that the size of the public-key has to be huge. Typically several hundred thousands of bits [2]. In 1991 a new public-key cryptosystem was presented by Gabidulin, Paramonov and Tretjakov. Its security relies on the problem of decoding codes in the rank metric [9]. It was a very promising system, since decoding algorithms in rank metric are exponential [14] . Therefore, this would allow a conceiver to choose public-keys of much smaller size. Unfortunately, since Gabidulin codes are strongly structured, Gibson showed that the public-key size had to be increased a lot. Hence it diminishes the practical interest of this system [10, 11].

The last developments in this field are not older than 2003 [16]. A new family of codes decodable in polynomial-time for rank metric was built. These codes are called *reducible rank codes*. This family of codes was used in a Niederreiter type cryptosystem.

In this paper, we design an efficient – in size and speed – public-key cryptosystem based on the family of reducible rank codes. Compared to the original system, we show that we can significantly increase the transmission rate of the system with a simple transformation using properties of the Rijndael S-boxes.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 218–229, 2004.

Moreover, we show that this new system can efficiently resist *message resend* attacks as well as *reaction attacks*, together with keeping a small public-key size. To achieve this goal, the designer only needs properties of rank metric and a *good* hash function.

2 Description of the Cryptosystem

In a first part, we present rank metric and some of its properties. In a second part, we introduce the problem of of decoding in rank metric. By the state of art this problem is difficult. The complexity of the best decoding algorithms show that it is theoretically possible to use public-keys of much smaller size than for PKCs (Public-Key Cryptosystems) based on problem of decoding in Hamming metric, like McEliece cryptosystem. In a third part, we describe the cryptosystem based on reducible rank codes as it was introduced in [16]. We only present the simplified version of the paper, since it is resistant to any kind of structural attacks as shown in the original paper.

2.1 Rank Metric

Rank metric was introduced in 1985 by E.M. Gabidulin [6]. Let $\mathrm{GF}(2^m)$ be the finite field with 2^m elements. Any element α of $\mathrm{GF}(2^m)$ can be written uniquely

$$\alpha = a_1\gamma_1 + \cdots + a_m\gamma_m,$$

where the $a_i \in \mathrm{GF}(2)$ and where $\gamma_1, \ldots, \gamma_m$ is a basis of $\mathrm{GF}(2^m)$ over $\mathrm{GF}(2)$. Any vector $\mathbf{c} = (c_1, \ldots, c_n)$ over $\mathrm{GF}(2^m)$ can be seen as an $m \times n$ matrix, whose ith column is the vector of size m corresponding to the expansion of the element c_i over the chosen basis. The rank of the vector c is by definition the rank of the obtained matrix. In the following it is denoted $\mathrm{Rk}(\mathbf{c} \mid \mathrm{GF}(2))$.

Consider n elements $g_1, \ldots, g_n$ of $\mathrm{GF}(2^m)$ which are linearly independent over $\mathrm{GF}(2)$. Consider the $k \times n$–matrix

$$G = \begin{pmatrix} g_1 & \cdots & g_n \\ g_1^{2^1} & \cdots & g_n^{2^1} \\ g_1^{2^2} & \cdots & g_n^{2^2} \\ \vdots & \ddots & \vdots \\ g_1^{2^{k-1}} & \cdots & g_n^{2^{k-1}} \end{pmatrix}. \tag{1}$$

Let $\mathcal{G}$ be the code generated by G, *i.e.*

$$\mathcal{G} = \{\mathbf{x}G \mid \mathbf{x} \in \mathrm{GF}(2^m)^k\}.$$

The code $\mathcal{G}$ is called *Gabidulin code*. Several polynomial-time algorithms correcting up to $t = \lfloor (n-k)/2 \rfloor$ errors in the rank metric were designed, [6, 7, 21, 17]. Our measurements of complexity will consider that we take the decoding algorithm described in [7], giving a decoding complexity of $\approx t^3 + (2n+m)t + k^2$ multiplications in $\mathrm{GF}(2^m)$.

2.2 Cryptosystems Based on Rank Codes

Since 1978, cryptosystems, identification schemes and even a signature scheme have been designed, the security of which relies on the problem of decoding linear codes in Hamming metric [13, 3]. This problem has been studied for a long time and some results about NP-Completeness have been obtained. The most recent can be found in [24]. In the same way, public-key cryptosystems and identification schemes have been designed on the problem of decoding linear codes in rank metric. Though the problem is believed to be hard, there is no existing proof of NP-Completeness [5, 9]. The problem of decoding in rank metric can be stated as such:

Input: A target vector $\mathbf{y}$ of length n over $\mathrm{GF}(2^m)$, a $k \times n$ matrix G of rank k over $\mathrm{GF}(2^m)$, and an integer t.
Decoding$(\mathbf{y}, G, t)$
Find, whenever it exists, a vector $\mathbf{x} \in GF(2^m)^k$*, and a vector* $\mathbf{e}$ *where* $Rk(\mathbf{e}|GF(2)) \leq t$ *such that* $\mathbf{y} = \mathbf{x}G + \mathbf{e}$*,*

The most efficient algorithms that solve this problem can be found in [14]. In this paper, Ourivski and Johannson present two algorithms that, given a linear code C of dimension k over $\mathrm{GF}(2^m)$, gives a solution to **Decoding**$(\mathbf{y}, G, t)$ with complexity

- *Minimum rank weight decoding*: $O\left((tm)^3 2^{(t-1)(k+1)}\right)$ binary operations.
- *Basis enumeration*: $O\left((k+t)^3 t^3 2^{(t-1)(m-t)}\right)$ binary operations.

Since these algorithms are strongly exponential in the dimension, minimum distance and extension degree of the codes, the **Decoding** problem in rank metric can be considered as a difficult problem. Therefore it could be suitable for designing PKCs.

Compared to the general decoding algorithms for Hamming metric, the decoding algorithms for rank metric have larger complexity for the same parameters. This implies that we should be able, at least theoretically, to design systems with smaller public-keys for a given security. This would give a nice solution to the major drawback of using cryptosystems based on linear codes that is to know the huge size of the public-key (several hundreds of thousands of bits for McEliece system to be secure against general decoding algorithms, [2]).

Unfortunately, in rank metric, the only families of known codes that are decodable in polynomial-time are constructed from Gabidulin codes. This family is very much structured and, although it is possible to hide the structure in some sense, Gibson showed that the public-key size had to be widely increased to prevent an attacker from breaking completely the system [10, 11].

Different alternatives were proposed to reduce further the public-key size [8, 15] with keeping a sufficient security. The general principle is the following:

- The conceiver chooses a generator matrix G of a code for which he knows a polynomial-time decoding algorithm up to some rank t.
- He scrambles the generator matrix G of the code in some way and then gets the matrix G_{pub}, that he publishes. He uses this matrix to encrypt the message in the same way as for the McEliece cryptosystem.

2.3 System Based on Reducible Rank Codes

Among the proposed PKC's based on rank metric, the newest, and maybe most original one uses the so called family of *reducible rank codes*. In the introductory paper a more general approach than ours is presented, [16]. However, since we are mainly interested in the optimality and the efficiency of the system we will only consider reducible rank codes of order 2, without the scrambling matrix. This does not diminish the security of the system.

– Let G be the matrix presented in (1). It generates a Gabidulin code of length n and dimension k. The code corrects $t = \lfloor (n-k)/2 \rfloor$ errors in rank metric in polynomial-time. Let A be a randomly chosen $k \times n$ matrix over GF(2^m). The designer forms the $2k \times 2n$ matrix G_{priv} such that

$$G_{priv} = \begin{pmatrix} G & A \\ 0 & G \end{pmatrix}.$$

– Then he picks up randomly a $2k \times 2k$ non-singular matrix S over GF(2^m) and a $2n \times 2n$ non-singular matrix P with coefficient over the base field GF(2).

The conceiver publishes the $2k \times 2n$-matrix

$$G_{pub} = SG_{priv}P.$$

The encryption–decryption procedure is the following.

– *Encryption:* Alice wants to send the information vector $\mathbf{x}$ of length $2k$ to Bob. She first chooses an error-vector $\mathbf{e}$ over GF(2^m) of length $2n$ and of rank t. Then she computes

$$\mathbf{y} = \mathbf{x}G_{pub} + \mathbf{e},$$

and sends $\mathbf{y}$ to Bob.

– *Decryption:* Since Bob knows the private key, he can compute

$$\mathbf{y}P^{-1} = \mathbf{x}S \begin{pmatrix} G & A \\ 0 & G \end{pmatrix} + \mathbf{e}P^{-1}.$$

For more convenience, let us write $\mathbf{x}' = \mathbf{x}S$, $\mathbf{e}' = \mathbf{e}P^{-1}$, and $\mathbf{y}' = \mathbf{y}P^{-1}$. The vector $\mathbf{x}'$ has length $2k$, thus $\mathbf{x}' = (\mathbf{x}'_1, \mathbf{x}'_2)$ where the $\mathbf{x}'_i$'s are of length k. Similarly we have $\mathbf{y}' = (\mathbf{y}'_1, \mathbf{y}'_2)$ where $\mathbf{y}'_i$ is of length n, and $\mathbf{e}' = (\mathbf{e}'_1, \mathbf{e}'_2)$. Then, Bob obtains the two following equations:

$$\begin{cases} \mathbf{y}'_1 = \mathbf{x}'_1 G + \mathbf{e}'_1, \\ \mathbf{y}'_2 = \mathbf{x}'_2 G + \mathbf{x}'_1 A + \mathbf{e}'_2. \end{cases}$$

The matrix P has coefficients in the base field GF(2). Therefore, multiplying by P^{-1} preserves the rank of the vectors. Since $\mathbf{e}$ has rank less than t, $\mathbf{e}' = \mathbf{e}P^{-1}$ has also rank less than t. As a consequence $\mathbf{e}'_1$ and $\mathbf{e}'_2$ have rank less than t. Hence:

1. By decoding $\mathbf{y}'_1$ in the code generated by G, Bob recovers $(\mathbf{x}'_1, \mathbf{e}'_1)$.
2. Knowing $\mathbf{x}'_1$ and A, he computes $\mathbf{y}_2 - \mathbf{x}'_1 A$. Then he decodes this vector in the code generated by G and recovers $(\mathbf{x}'_2, \mathbf{e}'_2)$.
3. Finally he gets the plaintext $\mathbf{x}$ by multiplying $(\mathbf{x}'_1, \mathbf{x}'_2)$ by S^{-1}.

Because of the small parameters used and because of the efficiency of the decoding algorithms for Gabidulin codes, this procedure is extremely fast.

3 The New System

In the original proposition, the authors use the Niederreiter form of the system, that is using the parity-check matrix. With this method, they can publish only the redundant part of the parity-check matrix, diminishing thus the key-size, which could not be done in the McEliece system without some information from the plaintext leaking out. They proposed a system taken over the field $\mathrm{GF}(2^{20})$ with a key-size of 16000 bits and a transmission rate of 0.55.

But there are two problems:

- The Niederreiter form of the system is not suitable against active attacks as we will mention further in the paper. Namely, a plaintext will always be encrypted into the same ciphertext, which is not the case when one uses the McEliece form, *i.e.* with the generator matrix. Therefore, there is absolutely no semantic security, since by making the difference of two received ciphertexts, an attacker can distinguish whenever the same message has been sent twice.
- The transmission rate is low. It is approximately equal to $(m+2n-t)t/(2m(n-k))$, which is much less than 1.

In the following, we show how to design a system based on reducible rank codes wich satisfies a *kind of* semantic security. We show how to increase significantly the transmission rate by designing a procedure that puts information in the error vector. By continuing our comparison with McEliece cryptosystem, adding information in the error-vector was done by Sendrier in [22], and the idea of rendering the scheme secure against message resend attacks can be found in [23].

In the modification of the system, we consider two things in addition to the standard parameters of the cryptosystem:

1. A hash function h taking as input $m \times 2n$ bits and returning $m \times 2k$ bits.
2. A procedure called $\mathcal{P}$ which takes as arguments a random vector r, an information vector $\widetilde{\mathbf{x}}$, and returns a vector of length $2n$ over $\mathrm{GF}(2^m)$ and of rank t. We want our procedure to satisfy three properties:
   (a) Be invertible on its image.
   (b) The procedures $\mathcal{P}$ and $\mathcal{P}^{-1}$ have to be computable in a time negligible compared to the time of the encryption-decryption of the system.
   (c) The procedure $\mathcal{P}$ must diffuse the randomness r *sufficiently* over its output vector.

With these tools, we design an encryption–decryption procedure slightly different from the original one:

- *Encryption:* Alice wants to encrypt the information vectors $\mathbf{x}, \widetilde{\mathbf{x}}$. She computes $\mathbf{y} = (\mathbf{x} + h(\mathcal{P}(r, \widetilde{\mathbf{x}})))G_{pub} + \mathcal{P}(r, \widetilde{\mathbf{x}})$. Finally she sends $\mathbf{y}$ to Bob.
- *Decryption:* Since the vector $\mathcal{P}(r, \widetilde{\mathbf{x}})$ has rank less than t, Bob recovers $\mathcal{P}(r, \widetilde{\mathbf{x}})$ and $\mathbf{x} + h(\mathcal{P}(r, \widetilde{\mathbf{x}}))$. Since $\mathcal{P}$ is invertible, he recovers $\mathbf{x}$ and $\widetilde{\mathbf{x}}$.

The security does not rely exactly on the decoding problem in rank metric. It is more related with the following problem, which can be stated as **Conditional Decoding**.

Input: A target vector $\mathbf{y}$ of length n over $\mathrm{GF}(2^m)$, a $k \times n$ matrix G of rank k over $\mathrm{GF}(2^m)$, and an integer t.
Conditional Decoding$(\mathbf{y}, G, t)$
Find, whenever it exists, a vector $\mathbf{x}$*, and a vector* $\mathbf{e}$ *where* $Rk(\mathbf{e}|GF(2)) \leq t$ *such that* $\mathbf{y} = (\mathbf{x} + h(\mathbf{e}))G + \mathbf{e}$*,*

We show that both problem are completely equivalent. This implies that the problem on which the security of our system is based is as difficult as the **Decoding** problem. Namely, suppose that we are given an algorithm $\mathcal{A}$ solving **Conditional Decoding**, *i.e.* $\mathcal{A}$ takes as input an instance $(\mathbf{y}, G, t)$ and returns in polynomial-time:

- *Failure*, if there is no solution to **Conditional Decoding**$(\mathbf{y}, G, t)$.
- If there is no failure it returns a pair $(\mathbf{x}, \mathbf{e})$, such that $\mathbf{y} = (\mathbf{x} + h(\mathbf{e}))G + \mathbf{e}$, and $\mathrm{Rk}(\mathbf{e} \mid \mathrm{GF}(2)) \leq t$.

Let $\mathcal{A}'$ be the following algorithm, taking as input $(\mathbf{y}, G, t)$ and returning:

- *Failure*, if $\mathcal{A}(\mathbf{y}, G, t)$ returns *Failure*.
- The pair $(\mathbf{x} - h(\mathbf{e}), \mathbf{e})$ if $\mathcal{A}(\mathbf{y}, G, t)$ returns $(\mathbf{x}, \mathbf{e})$.

The algorithm $\mathcal{A}'$ solves the **Decoding** problem in polynomial time for the instance $(\mathbf{y}, G, t)$. Namely,

- There is no solution to **Decoding**$(\mathbf{y}, G, t)$ if and only if there is no solution to **Conditional Decoding**$(\mathbf{y}, G, t)$. Indeed, if there were a solution $(\mathbf{c}_0, \mathbf{e}_0)$ to **Conditional Decoding**$(\mathbf{y}, G, t)$, then $(\mathbf{c}_0 + h(\mathbf{e}_0), \mathbf{e}_0)$, would be a solution to **Decoding**$(\mathbf{y}, G, t)$. The converse is easy to show.
- It is immediate to check that if there is a solution $(\mathbf{x}, \mathbf{e})$ to **Conditional Decoding**$(\mathbf{y}, G, t)$ then $(\mathbf{x} - h(\mathbf{e}), \mathbf{e})$ is a solution to **Decoding**$(\mathbf{y}, G, t)$, which is by construction of $\mathcal{A}'$ the value that the algorithm returns.

This show that our problems are both equivalent in terms of complexity.

3.1 The Procedure $\mathcal{P}$

Now we design the procedure $\mathcal{P}$. The goal of this procedure is to put information in an error-vector of rank t and of length $2n$. The number of possible error-vectors is equal to the number of vectors rank t, that is

$$\prod_{i=0}^{t-1} \frac{(2^m - 2^i)(2^{2n} - 2^i)}{2^t - 2^i}.$$

Since t is significantly less than $2n$ and m, this quantity can be approximated by $2^{(m+2n)t-t^2+1}$. Thus the maximal amount of information in bits that can be put on this vector is equal to $(m+2n)t-t^2+1$. An efficient encoding procedure to do this was described in [16], but here we are willing to keep r random bits in the error-vector.

- Let D be a binary $t \times t$ matrix of rank t.
- Let E_1 be a $(m-t)t$ matrix, and let E_2 be a $t(2n-t)$ matrix over GF(2).

The matrix

$$E = \begin{pmatrix} D & DE_2 \\ E_1 & E_1E_2 \end{pmatrix} \tag{2}$$

is a binary matrix of rank exactly t, and is thus the expansion of some vector over GF(2^m) of length $2n$ and of rank t. Information can be put in D, E_1 and E_2, as well as randomness.

An important point is to ensure a good diffusion of randomness in the matrix E. For example, suppose that the random positions are located in the matrix D, then the randomness of the error is located in a known subspace of dimension t. This fact could be used in the *message resend attacks* (cf. §3.3).

To avoid this problem, we propose to use the Rijndael S-box, [18, 19, 20]: it is an invertible function which takes in input a byte and return a byte in output. This S-box has good diffusion and non-linearity properties. The information bits and random bits must be spread in bytes in such a way that each byte contains at least one random bit. We apply the Rijndael S-box to each byte and then we put these in D, E_1 and E_2.

It is easy to check that recovering the sent message $\mathbf{x}'$ from this matrix is $O(t^3)$ binary operation and is thus negligible in complexity compared to the other procedures. Our procedure $\mathcal{P}$ satisfies thus all the requirements, that we demanded in the previous section.

3.2 Parameters of the Cryptosystem

By using the procedure $\mathcal{P}$ previously described, we increase the transmission rate of the system up to

$$\tau = \frac{2mk + (2n+m)t - r - t^2}{2mn} = k/n + \underbrace{\frac{(2n+m-t)t - r}{2mn}}_{\text{additional gain}}$$

The complexity of the encryption is:

- Computing $\mathbf{e} = \mathcal{P}(r, \widetilde{\mathbf{x}})$: Negligible compared to the other procedures since in $O(t^3)$ binary operations.
- Computing $(\mathbf{x} + h(\mathbf{e}))G_{pub}$: $\approx 4nk$ multiplications in GF(2^m).

The complexity of the decryption is:

- Multiplying by P^{-1}: $\approx 2n^2$ binary operations.
- Then one has to compute two decoding steps and a multiplication by A: $\approx 2t^3 + 2(2n + m)t + 2k^2 + nk$ multiplications in GF(2^m).
- Then multiplying by S^{-1}, supposed already precomputed: $4k^2$ multiplications in GF(2^m).

Therefore the overall complexity of the decryption is: $\approx t(2t^2 + 2(2n + m)) + k(6k + n)$ operations in the field GF(2^m).

3.3 Security of the Cryptosystem Based on Rank Codes

At the beginning of the section we showed that the problem on which we base the security of our system is equivalent to the problem of decoding in rank metric. However there might be various ways to attack the system.

Structural Attacks. They consist in attacking directly the public-key to break the system. In [16], it is discussed the resistance of the private key to any structural attacks. It is shown that it is not possible to break the system by recovering a decoder provided the parameters are well chosen. A security analysis implies moreover that the matrix G of the Gabidulin code can be published without loss of security. Indeed, one can pass from a Gabidulin code to another by changing the bases. Let G_{pub} be written under the form

$$G_{pub} = S \begin{pmatrix} G & A \\ 0 & G \end{pmatrix} P,$$

and let $G' = GU$, where U is non-singular with coefficients over GF(2) be a generator matrix of a Gabidulin code. We we have

$$G_{pub} = S \begin{pmatrix} G' & A \\ 0 & G' \end{pmatrix} P',$$

where

$$P' = \begin{pmatrix} U^{-1} & 0 \\ 0 & U^{-1} \end{pmatrix} P.$$

Therefore, publishing the matrix G of the private key does not give any additional information when one tries to cryptanalyse the system by attacking the public-key.

Decoding Attacks. For these attacks, an attacker intercepts a ciphertext and tries to recover the corresponding plaintext. The best known algorithms to achieve this goal were designed by Ourivski and Johansson and have complexity:

- *Minimum rank weight decoding*: $O\left((tm)^3 2^{(t-1)(2k+1)}\right)$ binary operations.
- *Basis enumeration*: $O\left((2k+t)^3 t^3 2^{(t-1)(m-t)}\right)$ binary operations.

Now we only discuss the resistance of the system against active eavesdropping. All basic attacks on our system can be naturally avoided, since there complexity is naturally exponential.

Message resend attack
In Hamming metric, it was presented by Berson in the case of McEliece cryptosystem [1]. It provides two different informations:

- An eavesdropper is able to know whenever the same message is sent twice because the Hamming weight of the difference of the encrypted message is very small.
- This knowledge gives information on the positions of errors, thus considerably diminishing the complexity of the decoding attack.

Suppose that the same message $\mathbf{x}$ is encrypted twice with our system. The attacker gets the two ciphertexts $\mathbf{y}_1 = (\mathbf{x} + h(\mathbf{e}_1))G_{pub} + \mathbf{e}_1$ and $\mathbf{y}_2 = (\mathbf{x} + h(\mathbf{e}_2))G_{pub} + \mathbf{e}_2$, where the $\mathbf{e}_i$'s contain some information and depend on r bits of randomness and on the transformation $\mathcal{P}$.

By computing the difference, he gets $\mathbf{y}_1 - \mathbf{y}_2 = (h(\mathbf{e}_1) - h(\mathbf{e}_2))G + \mathbf{e}_1 - \mathbf{e}_2$. Provided $h(\mathbf{e}_1) \neq h(\mathbf{e}_2)$, there is no way to distinguish this difference with the difference of two random ciphertexts. The probability that such an event occurs depends on the random parameter r. Even if it happens, distinguishing the ciphertexts does not enable one to recover the plaintext easily.

This analysis shows that the hash function introduces a *kind* of semantic security in our system. Another type of active attack that we consider are the reaction attacks.

Reaction attacks
Studying reaction attacks in case of rank metric is not so simple. Indeed, what plays the important role in the error vector is not a bit itself but a full vector space.

Reaction attacks can be described as follows: An attacker eavesdrops the channel and gets a ciphertext $\mathbf{y}$. He modifies a few bits of the ciphertext and then submits the new text to a decryption oracle. If the oracle does not reject the ciphertext it means that it is a valid ciphertext. Thus one obtains some information about the changed bits. This kind of attack is particularly adapted to Hamming metric, [12].

In rank metric, the basic idea would be in the same way to add some error on the ciphertext and then sees if this new message is accepted as a valid ciphertext. We can imagine the following attack:

- He picks up a vector $\mathbf{f}$ and compute $\mathbf{y}' = \mathbf{y} + \mathbf{f}$.
- He submits the new $\mathbf{y}'$ to the decryption oracle and sees its reaction: *Accept* or *Reject*.

By properties of rank metric, the oracle will accept $\mathbf{y}'$ as a valid ciphertext if and only if $\mathbf{f}+\mathbf{e}$ is of rank less than t, that is, since $\mathbf{e}=(e_1,\ldots,e_n)$ has rank t, if and only if every coordinate f_i of $\mathbf{f}$ is a linear combination of the e_j's. We can show that this attack can succeed in approximately $t2^{m-t}$ queries to the decryption oracle. Namely you can obtain a set of t elements forming a basis of the vector space spanned by the error-vector $\mathbf{e}$, and then you can decode and recover the plaintext in polynomial time. However note that in that case, the complexity of reaction attack is not linear but exponential in the size of the extension field. So the question how many queries can do the attacker to the oracle to complete a reasonable attack. We cannot answer to that question here but we see that if this is really problematic it is enough to increase the field size to secure the system.

3.4 Proposition of Parameters

We propose two different sets of parameters :
First set: $m=n=22$, $k=10$, $t=6$ and random parameter $r=80$.

- Size of the public-key : $22\times 24\times 20 = 10560$ bits if we consider only the redundant part of the matrix.
- Transmission rate of the system : ≈ 0.78.
- Security against the decoding attacks :
  1. *Minimum rank weight decoding*: $\approx 2^{126}$ binary operations.
  2. *Basis enumeration*: $\approx 2^{100}$ binary operations.

Note that in this case, for an equivalent security the public-key size of McEliece cryptosystem has to be of 700 kbits, that is 70 times larger [2]. The second set of parameters takes into account the necessary resistance to reaction attacks. Therefore we increase the size of the chosen finite field.

Second set: $m=60$, $n=20$, $k=10$, $t=5$ and random parameter $r=80$.

- Size of the public-key : $60\times 20\times 20 = 24000$ bits if we consider only the redundant part of the matrix.
- Transmission rate of the system : ≈ 0.65.
- Security against the decoding attacks :
  1. *Minimum rank weight decoding*: $\approx 2^{109}$ binary operations.
  2. *Basis enumeration*: $\approx 2^{281}$ binary operations.

This increases notably the size of the public-key but remains 30 times smaller than the public-key size of a secure McEliece cryptosystem. In the parameters we chose to propose a security of 2^{100}. For a security corresponding to 2^{80} binary operations, the key size can be made smaller.

4 Conclusion

From a cryptosystem based on rank metric, we designed a new one. It has the advantages of rank metric that is to know a small public-key size, and it is

resistant to different kinds of attacks. Namely, this system provides a kind of semantic security and can be rendered secure against reaction attacks and message resend attacks which are problematic if one uses systems such as McEliece cryptosystem.

References

1. T. A. Berson. Failure of the McEliece public-key cryptosystem under message resend and related-message attack. In *Advances in Cryptology, CRYPTO 1997*, Lecture Notes in Computer Science, pages 213–220, 1997.
2. A. Canteaut and N. Sendrier. Cryptanalysis of the original McEliece cryptosystem. In K. Ohta and D. Pei, editors, *Advances in Cryptology - ASIACRYPT'98*, number 1514 in LNCS, pages 187–199, 1998.
3. N. Courtois, M. Finiasz and N. Sendrier. How to achieve a McEliece-based signature scheme. In Colin Boyd, Editors, *Advances in Cryptology - ASIACRYPT'2001*, number 2248 in LNCS, pages 151–174,2001
4. F. Chabaud and J. Stern. The cryptographic security of the syndrome decoding problem for rank distance codes. In K. Kim and T. Matsumoto, editors, *Advances in Cryptology - ASIACRYPT '96*, volume 1163 of *LNCS*. Springer-Verlag, November 1996.
5. K. Chen. A new identification algorithm. In *Cryptographic policy and algorithms*, volume 1029, pages 244–249. Springer, 1996.
6. E. M. Gabidulin. Theory of codes with maximal rank distance. *Problems of Information Transmission*, 21:1–12, July 1985.
7. E. M. Gabidulin. A fast matrix decoding algorithm for rank-error correcting codes. In G. Cohen, S. Litsyn, A. Lobstein, and G. Zémor, editors, *Algebraic coding*, volume 573 of *LNCS*, pages 126–133. Springer-Verlag, 1991.
8. E. M. Gabidulin and A. V. Ourivski. Modified GPT PKC with right scrambler. In Daniel Augot and Claude Carlet, editors, *Proceedings of the 2nd International workshop on Coding and Cryptography, WCC 2001*, pages 233–242, 2001. ISBN Number : 2-761-1179-3.
9. E .M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their application in cryptology. *LNCS*, 547:482 – 489, 1991.
10. J. K. Gibson. Severely denting the Gabidulin version of the McEliece public-key cryptosystem. *Designs, Codes and Cryptography*, 6:37–45, 1995.
11. J. K. Gibson. The security of the Gabidulin public-key cryptosystem. In U. Maurer, editor, *EUROCRYPT'96*, pages 212–223, 1996.
12. C. Hall, I. Goldberg, and B. Schneier. Reaction attacks against several public-key cryptosystems. In *Proceedings of the 2nd International Conference on Information and Communication Security, ICICS'99*, number 1726 in LNCS, pages 2–12, 1999.
13. R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. Technical report, Jet Propulsion Lab. DSN Progress Report, 1978.
14. A. Ourivski and T. Johannson. New technique for decoding codes in the rank metric and its cryptography applications. *Problems of Information Transmission*, 38(3):237–246, September 2002.
15. A. V. Ourivski and E. M. Gabidulin. Column scrambler for the GPT cryptosystem. *Discrete Applied Mathematics*, 128(1):207–221, May 2003. Special issue of the second International Workshop on Coding and Cryptography (WCC2001).

16. A. V. Ourivski, E. M. Gabidulin, B. Honary, and B. Ammar. Reducible rank codes and their applications to cryptography. *IEEE Transactions on Information Theory*, 49(12):3289–3293, December 2003.
17. G. Richter and S. Plass. Fast Decoding of Rank-Codes with Rank Errors and Column Erasures In *Proceedings of ISIT 2004*, 2004.
18. J. Daemen, V. Rijmen *The Block Cipher Rijndael*, Smart Card Research and Applications, LNCS 1820, J.-J. Quisquater and B. Schneier, Eds., Springer-Verlag, 2000, pp. 288-296.
19. J. Daemen, V. Rijmen *Rijndael, the advanced encryption standard*, Dr. Dobb's Journal , Vol. 26, No. 3, March 2001, pp. 137–139.
20. `http://www.esat.kuleuven.ac.be/~rijmen/rijndael/rijndaelref.zip`
21. R. M. Roth. Maximum-Rank array codes and their application to crisscross error correction. *IEEE Transactions on Information Theory*, 37(2):328–336, March 1991.
22. N. Sendrier. Efficient generation of binary words of given weight. In C. Boyd, editor, *Cryptography and Coding, 5th IMA Conference, Circencester, UK*, LNCS, pages 184–187, December 1995.
23. H.-M. Sun. Enhancing the security of the McEliece public-key cryptosystem. *Journal of Information Science and Engeneering*, 16:799–812, 2000.
24. A. Vardy. The intractability of computing the minimum distance of a code. *IEEE Transactions on Information Theory*, 43(6):1757–1766, November 1997.

HEAD: Hybrid Encryption with Delegated Decryption Capability

Palash Sarkar

Cryptology Research Group, Applied Statistics Unit,
Indian Statistical Institute, 203, B.T. Road,
Kolkata, India 700108
palash@isical.ac.in

Abstract. We consider the problem of adding the functionality of delegated decryption on top of usual encryption schemes. An appropriate security model is described for such schemes. Our main contribution is to present a practical and efficient public key encryption scheme HEAD, which achieves the functionality of delegated decryption. The new scheme is obtained by combining bilinear map based techniques with Kurosawa-Matsuo modification of DHIES. The scheme HEAD is proved to be secure based on appropriate security assumptions on the components.

Keywords: public key encryption, DHIES, delegated decryption.

1 Introduction

Consider the following scenario: A busy corporate manager receives a large number of e-mails every day. For reasons of confidentiality, all e-mails are encrypted using the public key of the manager. Every day, the manager decrypts each e-mail, reads it and then delegates it to an assistant for proper handling. It turns out that most of the e-mails are of routine nature and could be handled directly by an assistant without the involvement of the manager.

Thus the problem boils down to the ability of the manager to delegate the decryption capability to an assistant. Clearly, the manager does not want any one assistant to be able to read *all* e-mails. The e-mails can be classified into categories and each category can be handled by one assistant. Any e-mail sent to the manager will have an unencrypted subject line mentioning the category and the body of the e-mail will be encrypted. An assistant who has been delegated to handle the particular category should be able to decrypt the e-mail and take proper action.

Let us consider how this can be achieved using standard encryption. One possibility is to write a "wrapper" around the mailer daemon, which performs a decryption using the secret key of the manager and then encrypts using the key of the assistant. There are several difficulties to this approach. First, if communication with the assistant is done using a symmetric cipher, then the

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 230–244, 2004.

manager needs to share a secret key with each of the assistant; if this is done using an asymmetric cipher, then the assistant has to obtain a public key/private key pair from a CA, which involves additional cost. Secondly, and perhaps more importantly, we have to address the question of automatic decryption using the manager's secret key. Secret keys are to be protected and so one would expect some kind of involvement by the manager during the decryption process. For example, if the manager is on vacation, then (s)he would not like to leave the secret key on the hard disk. This would stop the entire wrapper process during the manager's vacation period, which is an unacceptable situation.

From the above discussion, it is clear that trying to solve the problem with standard encryption technique introduces several difficulties. Thus, we are actually asking for a new primitive which has an additional functionality on top of standard cryptographic encryption. This functionality is that of delegated decryption. The question before the designer is to design a secure and efficient encryption scheme with the added functionality of delegated decryption.

In this paper, we provide a solution to this problem by modifying a known hybrid encryption scheme. We note that delegated decryption was briefly mentioned in [4] as a possible application of bilinear maps. However, a proper security model and a concrete scheme was not presented. To the best of our knowledge, the current work is the first one to propose a fully functional, efficient and provably secure encryption scheme with delegated decryption capability.

DHIES [2] is a hybrid encryption scheme which is present in draft standards of IEEE P1636a and ANSI X9.63 [3]. It combines an ElGamal type encryption with a symmetric encryption and a message authentication code (MAC) generation scheme. DHIES is an efficient encryption scheme and has been proved to be secure under the oracle Diffie-Hellman assumption; chosen plaintext security for symmetric encryption and unforgeable chosen message security for the MAC scheme. In a recent work, Kurosawa and Matsuo [10] showed that the MAC component can be removed from DHIES. Instead, they require the symmetric encryption scheme to satisfy chosen ciphertext security.

We also modify DHIES to achieve delegated decryption functionality. Following [10], we do away with the MAC component and use only a symmetric encryption scheme. Our main innovation, however, is in the replacement of the public key part of DHIES. We use a bilinear pairing based technique to achieve the required functionality of delegated decryption. The new encryption scheme is called HEAD. As a result of introducing a bilinear based public key part, we have to introduce a new hard problem, which we call the oracle bilinear Diffie-Hellman problem. This is an adaptation of the oracle Diffie-Hellman problem introduced in [2].

The scheme HEAD is proved to be secure. For this, we introduce an appropriate notion of security for an encryption scheme with delegated decryption capability. In the current version of the paper, we do not consider insider attacks, i.e., attack by a collusion of assistants. It is not difficult to modify our security model and proof technique to prove resistance against insider attacks. This will

be presented in a future communication along with several other applications of the basic primitive.

For the security proof to work, we need an extension of the usual notion of chosen ciphertext security for the symmetric encryption component. Though we have not been able to prove that this new notion is equivalent to the usual notion, we believe that the symmetric encryption schemes in [8, 9] satisfy the new notion. Settling this is a possible future research problem.

HEAD is an efficient enciphering scheme. For encryption and decryption, we require one pairing each plus symmetric key encryption and decryption. Compared to the Kurosawa-Matsuo variant of DHIES, we have replaced the exponentiations by pairings. While this leads to a slowdown, we are able to achieve the added functionality of delegated decryption.

Related Primitives: There are several primitives which have functionality similar to that of delegated decryption. We mention these below.

In an earlier work [12], a different (and somewhat weaker) form of delegated decryption was considered. In the setting of [12], a user has a public key and a private key to be used for the dual purpose of decryption and signing. He (or she) can delegate the decryption capability to an assistant, while retaining (and not delegating) the capability to sign a message. The assistant who receives the delegated decryption key can decrypt *all* messages of the user. In contrast, in our setting, decryption keys can be selectively assigned to different assistants. Since the main secret key of the user is not given away, the signing ability is retained by the master user.

In proxy signature schemes, the signing capability is delegated to an assistant, i.e., the assistant can sign a message on behalf of a user. This was introduced by Mambo et al in [11] and has later been studied by a number of authors. There is also a recent proposal on bilinear map based proxy signature scheme [13]. See [1] for a bibliography on proxy signatures. In both proxy signatures and delegated decryption, the assistant is given a secret key which is derived from the secret key of the master user. On the other hand, the encryption method that we propose is a hybrid encryption method. Thus, even though proxy signature and delegated decryption (with hybrid encryption) have similarities, it is not clear how to obtain one primitive from the other.

In a recent work, Boneh et al [5] have introduced a new primitive – that of public key encryption with keyword search. This primitive allows a third party (i.e., neither Alice nor Bob) to search for selective keywords in an encrypted message on behalf of Alice. However, this third party does not learn anything more about the e-mail; in particular it cannot decrypt the e-mail. On the other hand, in delegated decryption, we require the third party to be able to decrypt selected e-mails on behalf of Alice. This, of course, trivially implies the ability to look for keywords in the e-mail. Note that this feature might not always be desirable and the applications of encrypted keyword search and delegated decryption are really different.

Lastly, we would like to mention the primitive of broadcast encryption. In this primitive, there are several private keys corresponding to a single encryption

key. The distribution centre encrypts a message using the encryption key and broadcasts the message. The individual users with proper decryption keys can decrypt the message. Note that in this set-up all users are able to decrypt the message. This is a special case of our protocol. In fact, our protocol can also be used for broadcast encryption, with the added functionality of allowing different sets of users to decrypt different messages.

2 Basics

This section consists of several subsections describing models of system components and hardness assumptions. We start by presenting the model of asymmetric encryption possessing the capability for delegating decryption ability.

2.1 Asymmetric Encryption with Delegated Decryption Capability

A usual asymmetric encryption scheme asym is a tuple
$\mathsf{asym} = (\mathcal{M}, \mathcal{C}, \mathcal{SK}, \mathcal{PK}, \mathsf{asym.keygen}, \mathsf{asym.enc}, \mathsf{asym.dec})$ where

- $\mathcal{M}$ and $\mathcal{C}$ are respectively the message and cipher spaces;
- $\mathcal{SK}$ and $\mathcal{PK}$ are respectively the secret and public key spaces;
- $\mathsf{asym.enc}(\mathsf{pk}, M)$ is the encryption algorithm which takes as input a key $\mathsf{pk} \in \mathcal{PK}$ and
  a message $M \in \mathcal{M}$ and produces a cipher $C \in \mathcal{C}$.
- $\mathsf{asym.dec}(\mathsf{sk}, C)$ is the decryption algorithm which takes as input a key $\mathsf{sk} \in \mathcal{SK}$ and a cipher $C \in \mathcal{C}$ and either returns Bad or produces a message $M \in \mathcal{M}$ such that $\mathsf{asym.dec}(\mathsf{sk}, \mathsf{asym.enc}(\mathsf{pk}, M)) = M$.

A matching pair of private-public key $(\mathsf{sk}, \mathsf{pk})$ is produced by invoking the key generation algorithm $\mathsf{asym.keygen}$ on the security parameter. The encryption algorithm and the key generation algorithms are randomized algorithm. Further, the run-time of all the algorithms is polynomial time in the security parameter. This is usually enforced by providing the unary encoding of the security parameter as an additional input to all the algorithms.

For an asymmetric encryption scheme possessing delegated decryption capability, the following modifications are required. There is a master user and a set of assistants each of whom are identified by a binary string called a subject line.

1. There are l subject lines $\mathsf{SL}_1, \ldots, \mathsf{SL}_l$.
2. Corresponding to a subject line SL_i, the master user generates a secret information Q_i.
3. The secret Q_i is transmitted securely to the i-th assistant.
4. The encryption algorithm takes an additional input SL_i.
5. The decryption algorithm takes an additional input Q_i provided by the i-th assistant.

The idea is that the i-th assistant and the master user can decrypt the message which has been encrypted using subject line SL_i. Nobody else should be able

to decrypt the message. Below we formalize this as part of the security requirement. First, we describe the usual notion of security for asymmetric encryption and then describe the modification required for handling delegated decryption.

The usual notion of security for asymmetric encryption is as follows. The adversary runs in two stages – the find stage followed by the guess stage. In both stages, the adversary has access to a decryption oracle, which is the decryption algorithm instantiated by a randomly chosen secret (i.e., unknown to the adversary) key. In both stages, the adversary can query the decryption oracle with messages and receive the corresponding ciphertexts. At the end of the find stage, the adversary outputs two messages (x_0, x_1). A bit $b \in \{0, 1\}$ is selected at random and x_b is encrypted using the encryption oracle. The adversary then starts the guess stage. In the guess stage, the adversary is not allowed to query the decryption oracle on the target y. At the end of the guess stage, it outputs a bit b'. The adversary's advantage in breaking the system is defined to be $2|\Pr[b = b'] - 1/2|$.

In case of delegated decryption capability, the modifications required are as follows:

1. The find stage remains as above.
2. The target generation is changed: The message x_b is encrypted using *all* the l subject lines and the l targets $T_1, \ldots, T_l$ are given to the adversary.
3. The guess stage remains as above with the (natural) restriction that the adversary cannot query the decryption oracle on any of the l targets $T_1, \ldots, T_l$.

At the end, the adversary outputs b' and the advantage of the adversary is defined as above. The encryption algorithm being a randomized algorithm ensures that with very high probability the targets $T_1, \ldots, T_l$ are distinct; the probability that two are equal being determined by the birthday paradox.

Another kind of possible attack on a scheme with delegated decryption capability is insider attack, i.e., an attack by a collusion of assistants. For this we have to consider the possibility that the adversary can (adaptively) obtain the keys of some of the assistants. We have to prove that this does not provide the adversary with significant advantange in winning the adversarial game. It is possible to extend our proof technique to obtain such a proof. For the proof to work we require a separate hardness assumption on the map-to-point funtion $M()$ used in the protocol. The details of the model and the proof will be presented in a future communication.

2.2 Symmetric Encryption

A symmetric encryption scheme sym is a tuple
$\mathsf{sym} = (\mathcal{M}, \mathcal{C}, \mathcal{K}, \mathsf{sym.keygen}, \mathsf{sym.enc}, \mathsf{sym.dec})$ where

- $\mathcal{M}$, $\mathcal{C}$ and $\mathcal{K}$ are respectively the message, cipher and key spaces;
- $\mathsf{sym.enc}(K, M)$ is the encryption algorithm which takes as input a key $K \in \mathcal{K}$ and a message $M \in \mathcal{M}$ and produces a cipher $C \in \mathcal{C}$.

- $\mathsf{sym.dec}(K, C)$ is the decryption algorithm which takes as input a key $K \in \mathcal{K}$ and a cipher $C \in \mathcal{C}$ and either returns Bad or produces a message $M \in \mathcal{M}$ such that $\mathsf{sym.dec}(K, \mathsf{sym.enc}(K, M)) = M$.

The algorithm $\mathsf{sym.keygen}()$ is a randomized algorithm which takes as input a security parameter and returns a key for the system. In our application, the key generation algorithm will return a key uniformly at random from the set of all possible keys.

We consider chosen ciphertext security for sym. The usual model of security is extended in the following manner. First l keys $K_1, \ldots, K_l \in \{0,1\}^{k_S}$ are chosen randomly. An adversary $\mathcal{A}$ for sym is given access to l decryption oracles $\mathcal{D}_{K_i}(y) = \mathsf{sym.dec}(K_i, y)$. The adversary $\mathcal{A}$ runs in two stages – the find stage followed by the guess stage.

In the find stage, $\mathcal{A}$ can make arbitrary queries to any of $\mathcal{D}_{K_i}()$. At the end of the find stage, $\mathcal{A}$ produces two messages x_0 and x_1.

A bit b is randomly chosen and x_b is encrypted using the keys $K_1, \ldots, K_l$ to obtain l ciphertexts $y_1, \ldots, y_l$. These l ciphertexts are given to $\mathcal{A}$.

Adversary $\mathcal{A}$ then enters the guess stage. In the guess stage, the adversary is allowed access to the decryption oracles with the only (natural) restriction that $\mathcal{D}_{K_i}()$ is not queried with y_i.

Finally, $\mathcal{A}$ produces a bit b'. Formally, we define

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{sym}} = 2\left|\Pr[b = b'] - \frac{1}{2}\right| \tag{1}$$

to be the advantage that $\mathcal{A}$ has in breaking sym. The quantity $\mathsf{Adv}^{\mathsf{sym}}(t, q)$ is defined to be the maximum of $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{sym}}$, where the maximum is taken over all adversaries $\mathcal{A}$ running in time at most t and making a total of at most q queries to its decryption oracles. Note that we do not allow $\mathcal{A}$ access to encryption oracles, since this feature will not be required in our security proof.

In the usual notion of chosen ciphertext security $l = 1$. It is clear that any adversary which can break a scheme given l_1 targets can also break it given $l_2 > l_1$ targets. The converse, however, is not clear. On the other hand, it is unlikely that for secure symmetric encryption schemes (as in [8, 9]), choosing l to be a small value (around 100 or so) will provide additional information to an adversary. The exact theoretical status of the problem appears to be an interesting research question.

2.3 Cryptographic Bilinear Map

Let $G_1 = \langle P \rangle$ and G_2 be two groups of the same prime order p. We view G_1 as an additive group and G_2 as a multiplicative group. A mapping $e : G_1 \times G_1 \to G_2$ satisfying the following properties is called a cryptographic bilinear map:

Bilinearity	: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1$ and $a, b \in Z_p$.
Non-degeneracy	: If $G_1 = \langle P \rangle$, then $G_2 = \langle e(P, P) \rangle$.
Computability	: There exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G_1$.

Modified Weil Pairing [4] and Tate Pairing [6, 7] are examples of cryptographic bilinear maps. Informally, the bilinear Diffie-Hellman (BDH) problem is the following: Given P, aP, bP and cP compute $e(P,P)^{abc}$. Again, informally, the BDH assumption is that there is no "efficient algorithm" to solve the BDH problem. We do not formalize the BDH problem or the BDH assumption, since this will not be required by us. Instead we will formalize a variant of the BDH problem, whose hardness will be the basis for our security proof.

2.4 Hardness Assumption

Let $G_1 = \langle P \rangle$, G_2 be groups of prime order p and $e(,)$ be as defined in Section 2.3. Following Bellare and Rogaway [2], we introduce a new hardness assumption involving the bilinear Diffie-Hellman problem, a hash function $H : G_1 \times G_2 \rightarrow \{0,1\}^k$ and a map-to-point function $M : \{0,1\}^* \rightarrow G_1$. We call the new assumption the oracle bilinear decision Diffie-Hellman (OBDH) assumption. The OBDH problem is formally defined as follows.

Instance	: $(P, aP, bP, \mathsf{SL}, \mathsf{str})$, where $\mathsf{SL} \in \{0,1\}^*$, $a, b \in Z_p$ and $\mathsf{str} \in \{0,1\}^k$.
Oracle	: $\mathcal{H}_a(X, Y)$, with $X, Y \in G_1$. When invoked with $(a_1 P, a_2 P)$ returns $H(a_1 P, e(a_1 P, a_2 P)^a)$.
Oracle Restriction	: Cannot query $\mathcal{H}_a(,)$ on $(bP, M(\mathsf{SL}))$.
Task	: Determine whether $\mathsf{str} = H(bP, e(bP, M(\mathsf{SL}))^a)$ or str is random.

Notes:

1. The map from $G_1 \times G_1$ to G_2 defined by $(Q, R) \rightarrow e(Q, R)$ is not a bijection; for any $c \in Z_p$, the pairs (Q, R) and $(cQ, c^{-1}R)$ both map to $e(Q, R)$. To avoid this problem, we create a bijection from $G_1 \times G_1$ to $G_1 \times G_2$ by $(Q, R) \rightarrow (Q, e(Q, R))$. This bijection is used in the definition of the OBDH problem.
2. The function $H()$ will be instantiated by a cryptographic hash function such as SHA-256. The input to $H()$ is the pair $(Q, e(Q, R))$. Providing Q in "raw form" as input to $H()$ is expected to destroy algebraic properties.

Informally, the OBDH assumption states that there is no "efficient algorithm" to solve the OBDH problem. We formalize this statement in a manner which will be useful for our reduction based security proof. We consider algorithms to solve OBDH in the following manner. Any algorithm $\mathcal{A}$ takes as input an instance $(P, aP, bP, \mathsf{SL}, \mathsf{str})$ of OBDH and produces as output either zero or one. The advantage of an algorithm $\mathcal{A}$ in solving OBDH is formally defined in the following manner.

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{obdh}} = |\mathsf{Pr}[\mathcal{A} \text{ outputs } 1 | E_1] - \mathsf{Pr}[\mathcal{A} \text{ outputs } 1 | E_2]| \tag{2}$$

where E_1 is the event that $\mathsf{str} = H(bP, e(bP, M(\mathsf{SL}))^a)$ and E_2 is the event that str is random. The quantity $\mathsf{Adv}^{\mathsf{obdh}}(t, q)$ denotes the maximum of $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{obdh}}$

where the maximum is taken over all adversaries running in time at most t and making at most q queries to the oracle $\mathcal{H}_a(,)$.

The intuitive justification for OBDH assumption in the context of decision Diffie-Hellman problem has been given in [2]. Similar arguments hold here and hence we do not repeat them. The basic idea is that if $H()$ is a "cryptographic" hash function, then access to the oracle (subject to the oracle restriction) will not help the adversary in solving the OBDH problem.

3 Description of HEAD

There is a master user and a set of assistants. Each encrypted message comes with a subject line. An assistant who has been delegated to deal with the particular subject line can decrypt the message on behalf of the master user.

3.1 Basic Components and Set-Up

The following components are required.

1. Let $G_1 = \langle P \rangle$, G_2 be groups of order p and $e(,)$ be as defined in Section 2.3.
2. A symmetric encryption scheme $\mathsf{sym} = (\{0,1\}^*, \mathcal{C}, \{0,1\}^{k_S}, \mathsf{sym.enc}, \mathsf{sym.dec})$.
3. A hash function $H : G_1 \times G_2 \to \{0,1\}^{k_S}$.
4. A map-to-point function $M : \{0,1\}^* \to G_1$.

Note that the length of the symmetric key is k_S. We assume the OBDH problem to be hard for (G_1, G_2, e, H, M) in the sense defined in Section 2.4.

The master user generates a random $a \in Z_p$ and computes $\mathsf{Pub} = aP$. The value a is kept secret while Pub is made public. Only the master user knows the value a. In a PKI set-up, we assume that the public information Pub is signed by the certifying authority (CA).

3.2 Encryption

Suppose a user wants to send a message x to the master user. The encryption algorithm is as follows.

Algorithm $\mathsf{HEAD.enc}(x)$
1. generate random $r \in Z_p$ and compute rP;
2. choose a subject line $\mathsf{SL} \in \{0,1\}^*$;
3. compute $\mathsf{sym.key} = H(rP, e(\mathsf{Pub}, M(\mathsf{SL}))^r)$;
4. compute $y = \mathsf{sym.enc}(\mathsf{sym.key}, x)$;
5. the ciphertext is (rP, y, SL);

end algorithm.

The message x is encrypted with subject line SL. The assistant (or the master user) who decrypts the ciphertext must possess a secret key corresponding to the subject line SL. There may be a requirement that for certain messages which are marked "personal" only the master user should be able to decrypt. In this case, the master user acts as his own assistant.

3.3 Decryption

Suppose $(X = rP, y, \mathsf{SL})$ is received, where $X \in G_1$. Decryption by an assistant is as follows. We assume that the assistant possesses a secret key corresponding to the subject line SL. This key is privately given to him or her by the master user and is defined as $Q_{\mathsf{SL}} = aM(\mathsf{SL})$. This key may be given to the assistant after the message is received or it may have been given earlier. The exact procedure depends on the application. We discuss more about it in Section 3.4.

Algorithm $\mathsf{HEAD.dec}(X, y, \mathsf{SL})$
1. compute $\mathsf{sym.key} = H(X, e(X, Q_{\mathsf{SL}}))$;
2. compute $x = \mathsf{sym.dec}(\mathsf{sym.key}, y)$;
3. return x;

end algorithm.

Note that if $\mathsf{sym.dec}(\mathsf{sym.key}, y)$ returns Bad then $\mathsf{HEAD.dec}$ also returns Bad. The decryption succeeds because of the following computation:

$$e(X, Q_{\mathsf{SL}}) = e(rP, aM(\mathsf{SL})) = e(aP, M(\mathsf{SL}))^r = e(\mathsf{Pub}, M(\mathsf{SL}))^r.$$

The second step involves the bilinearity property of $e(,)$.

3.4 Applications

The first thing to observe is that the only information signed by the CA is Pub. The subject lines or their map-to-point images are not signed by the CA. Thus the involvement with the CA is one-time and subject lines can be created as and when required.

There is a similarity between HEAD and the identity based encryption scheme of Boneh and Franklin [4]. As in the case of identity, the subject line can be any string. This provides a great degree of flexibility in designing applications.

There might be a fixed set of assistants who are defined using certain keywords. These keywords could be their IDs or keywords defining their roles. One issue in such a set-up is that of key escrow. If an employee leaves the company or is redeployed for other work, (s)he should not be able to decrypt any further messages corresponding to his or her earlier role. This can be achieved by defining a subject line to be a pair of strings consisting of the role and the assistant ID. If a new assistant takes over the role, the subject line changes and the old assistant should be unable to decrypt any new messages. Additionally, a time stamping mechanism can be used.

Another interesting application is to allow a more dynamic form of delegation. Note that during encryption, there is no involvement of any assistant or the master user. Thus, the sender can choose *any* subject line and use it to encrypt the message. The encrypted message which is received by the master user cannot be decrypted by any of the assistants at this point, since none of them have the secret key corresponding to the subject line which has been used. The master user can read the message and delegate the handling of the message to one of the assistants. To do this, he creates the secret key corresponding to the subject

line and gives the key to the designated assistant. This allows for a dynamic delegation scheme.

Both fixed and dynamic delegation can be used. The fixed delegation corresponds to routine messages, while dynamic delegation corresponds to new messages that might be received. The security model described in Section 2.1 considers only fixed delegation, i.e., a set of pre-defined subject lines. In Section 4, we define an easy extension of the model to allow dynamic delegation.

4 Security Reduction

The security model for asymmetric encryption with delegated decryption capability is given in Section 2.1. This model provides for use of l pre-defined subject lines. For the case of HEAD, the subject lines can be both pre-defined and dynamic as explained in Section 3.4. In the security proof below we allow the adversary to make decryption queries with arbitrary subject lines. However, the targets given to the adversary before the start of the guess stage are encrypted using the l pre-defined subject lines.

In this section, we prove the security of HEAD, i.e., we show that breaking HEAD implies either solving OBDH, or breaking sym in the sense described in Section 2.2. Suppose the system components and the public parameters have been fixed as in Section 3.1. Additionally, there are l pre-defined subject lines $\mathsf{SL}_1, \ldots, \mathsf{SL}_l$.

We model the adversarial behaviour described in Section 2.1 as a game in the following manner. For $0 \leq i \leq l$, we define the game $\mathcal{G}_i$ as follows. The find and guess stages of Section 2.1 remain unchanged. At the end of the find stage the adversary generates (x_0, x_1) as usual. The target generation for game $\mathcal{G}_i$ is changed as follows:

choose a random bit $b \in \{0, 1\}$.
for $j = 1$ to i do
  randomly choose $r_j \in Z_p$ and form $X_j = r_j P$;
  randomly choose $\mathsf{sym.key} \in \{0,1\}^{k_S}$;
  set $y_j = \mathsf{sym.enc}(\mathsf{sym.key}, x_b)$;
  set $T_j = (X_j, y_j, \mathsf{SL}_j)$;
end for;
for $j = i + 1$ to l do
  generate T_j from x_b using SL_j and Pub as defined by HEAD.enc;
end for;
output targets $(T_1, \ldots, T_l)$;

Note that the keys for sym are independently and randomly chosen for the targets $T_1, \ldots, T_i$.

At the end of the guess stage, the adversary outputs b'. We define the output of $\mathcal{G}_i$ to be 1 if $b = b'$, else we define it to be 0. The notation $\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1]$ denotes the probability that the game $\mathcal{G}_i$ with adversary $\mathcal{A}$ outputs one. We formally define

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HEAD}} = 2\,|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2| \tag{3}$$

to be the advantage that an adversary $\mathcal{A}$ has in breaking HEAD. We define $\mathsf{Adv}^{\mathsf{HEAD}}(t, q)$ to be maximum of $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{HEAD}}$ where the maximum is taken over all adversaries running in time at most t and making at most q queries to its decryption oracle.

Lemma 1. *Let $\mathcal{A}$ be an adversary for* HEAD *which runs in time t and makes q queries to its decryption oracle. Then for all $0 \leq i \leq l-1$,*

$$|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \leq \frac{p}{p-q}\mathsf{Adv}^{\mathsf{obdh}}(t, q).$$

Proof. Our proof is a reduction. We show that if $\mathcal{A}$ can "distinguish" between $\mathcal{G}_i$ and $\mathcal{G}_{i+1}$, then it is possible to construct an algorithm $\mathcal{C}$ to solve the OBDH problem.

Let $(aP, bP, \mathsf{SL}, \mathsf{str})$ be an instance of the OBDH problem with an associated oracle $\mathcal{H}_a(,)$. Algorithm $\mathcal{C}$ first constructs an instance of HEAD in the following manner. For $j = 1, \ldots, l$ with $j \neq i+1$, randomly choose $\mathsf{SL}_j \in \{0,1\}^*$. Set $\mathsf{SL}_{i+1} = \mathsf{SL}$ and $\mathsf{Pub} = aP$. The public information $(\mathsf{Pub}, \mathsf{SL}_1, \ldots, \mathsf{SL}_l)$ is provided to $\mathcal{A}$.

The find stage decryption queries are simulated in the following manner: Suppose (rP, y, SL') is a decryption query. Invoke $\mathcal{H}_a(,)$ on $(rP, M(\mathsf{SL}'))$ to obtain $\mathsf{str}' = H(rP, e(rP, M(\mathsf{SL}'))^a)$. Decrypt y using str' and return the answer to $\mathcal{A}$. Note that if $\mathsf{SL}' = \mathsf{SL}$, then with certain probability $r = b$ and due to the oracle restriction, algorithm $\mathcal{C}$ cannot invoke $\mathcal{H}_a$ on $(rP = bP, M(\mathsf{SL}))$ and thus fail to answer the query. In this case, algorithm $\mathcal{C}$ outputs a random bit and exits. Let Fail be the event that $\mathcal{C}$ fails to answer some find stage decryption query. The probability that $\mathcal{C}$ fails for any particular query is the probability that for that query $r = b$ and this probability is equal to $1/|Z_p| = 1/p$. Since $\mathcal{A}$ makes a total of q queries, we have $\mathsf{Pr}[\mathsf{Fail}] = q/p$.

The targets are generated in the following manner. First b is chosen randomly from $\{0,1\}$.

For $1 \leq j \leq i$, randomly generate $r_j \in Z_p$ and form $X_j = r_j P$. Randomly choose $K_1, \ldots, K_i \in \{0,1\}^{k_S}$. Encrypt x_b with $K_1, \ldots, K_i$ to obtain $y_1, \ldots, y_i$ respectively. Set $T_j = (X_j, y_j, \mathsf{SL}_j)$, for $1 \leq j \leq i$.

For $j = i+1$, set $X_{i+1} = bP$ and encrypt x_b with str to obtain y_{i+1}. Set $T_{i+1} = (bP, y_{i+1}, \mathsf{SL}_{i+1} = \mathsf{SL})$.

For $i+2 \leq j \leq l$, encrypt x_b properly using HEAD.enc on Pub and subject line SL_j to obtain target T_j. All the targets $T_1, \ldots, T_l$ are given to the adversary.

Simulation of the decryption queries in the guess stage is as follows: Let the submitted query be (X, y, SL'). There are several cases:

Case 1: $X = X_j$ and $\mathsf{SL}' = \mathsf{SL}_j$ for some $j \in \{1, \ldots, i\}$: We must have $y \neq y_j$, as otherwise $(X, y, \mathsf{SL}') = T_j$, which is not allowed. In this case, use K_j to decrypt y and provide the answer to $\mathcal{A}$.

Case 2: $X = X_{i+1} = bP$ and $\mathsf{SL}' = \mathsf{SL}_{i+1} = \mathsf{SL}$: Again $y \neq y_{i+1}$ and we use str to decrypt y and provide the answer to $\mathcal{A}$.

Case 3: Either ($X = X_j$ and $\mathsf{SL}' = \mathsf{SL}_j$ for some $j \in \{i+2, \ldots, l\}$) or $(X, \mathsf{SL}') \neq (X_j, \mathsf{SL}_j)$ for any $j \in \{1, \ldots, l\}$: In this case obtain str' by querying $\mathcal{H}_a(,)$ on $(X, M(\mathsf{SL}'))$. Use str' to decrypt y and provide the answer to $\mathcal{A}$.

This completes the description of the simulation of adversary $\mathcal{A}$ by algorithm $\mathcal{C}$. Let E_1 be the event that $\mathsf{str} = H(bP, e(bP, M(\mathsf{SL}))^a)$ and E_2 be the event that str is random. Suppose that E_1 occurs. Then there are two possibilities – either Fail occurs or Fail does not occur. In the first case, $\mathcal{C}$ outputs a random bit and in the second case, $\mathcal{C}$ runs $\mathcal{A}$ on game $\mathcal{G}_i$. Thus,

$$\mathsf{Pr}[\mathcal{C} = 1|E_1] = \frac{1}{2} \times \frac{q}{p} + \mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] \times \left(1 - \frac{q}{p}\right).$$

By a similar argument, if E_2 occurs, we have

$$\mathsf{Pr}[\mathcal{C} = 1|E_2] = \frac{1}{2} \times \frac{q}{p} + \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1] \times \left(1 - \frac{q}{p}\right).$$

Combining the above two equations we have,

$$|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| = \frac{p}{p-q} \times |\mathsf{Pr}[\mathcal{C} = 1|E_1] - \mathsf{Pr}[\mathcal{C} = 1|E_2]|. \quad (4)$$

Since adversary $\mathcal{A}$ for HEAD makes at most q decryption queries, algorithm $\mathcal{C}$ for OBDH also makes at most q oracle queries. Further, since adversary $\mathcal{A}$ runs in time t, algorithm $\mathcal{C}$ also runs in time t. Thus we have, $|\mathsf{Pr}[\mathcal{C} = 1|E_1] - \mathsf{Pr}[\mathcal{C} = 1|E_2]| \leq \mathsf{Adv}^{\mathsf{obdh}}(t, q)$. Substituting this in (4), we get the desired result. □

We now consider the game $\mathcal{G}_l$. For this game, the targets $T_1, \ldots, T_l$ are generated using random keys for sym and these keys have no relation to the subject lines or the public key part (rP). Thus the only way the adversary can win this game is to break sym. We formalize this below.

Lemma 2. *Let $\mathcal{A}'$ be an adversary for $\mathcal{G}_l$ running in time t and making q queries to its decryption orable. Then* $2|\mathsf{Pr}[\mathcal{A}'(\mathcal{G}_l) = 1] - 1/2| \leq \mathsf{Adv}^{\mathsf{sym}}(t, q)$.

Proof. We bound the success probability of $\mathcal{A}'$ in winning game $\mathcal{G}_l$. For this, we construct an adversary $\mathcal{A}$ for breaking sym in the sense described in Section 2.2.

The adversary $\mathcal{A}$ is given l decryption oracles $\mathcal{D}_{K_1}, \ldots, \mathcal{D}_{K_l}$ corresponding to randomly chosen keys $K_1, \ldots, K_l \in \{0,1\}^{k_S}$. The first task of $\mathcal{A}$ is to set up an instance of HEAD. To do this $\mathcal{A}$ chooses a random $a \in Z_p$ and forms $\mathsf{Pub} = aP$. It then chooses l subject lines $\mathsf{SL}_1, \ldots, \mathsf{SL}_l$ and gives $(\mathsf{Pub}, \mathsf{SL}_1, \ldots, \mathsf{SL}_l)$ to $\mathcal{A}'$.

Let $(X = rP, y, \mathsf{SL})$ be a find stage decryption query by $\mathcal{A}'$. To answer this query $\mathcal{A}$ forms

$$\begin{aligned} \mathsf{str} &= H(X, e(X, M(\mathsf{SL}))^a) = H(rP, e(rP, M(\mathsf{SL}))^a) \\ &= H(rP, e(aP, M(\mathsf{SL}))^r) = H(rP, e(\mathsf{Pub}, M(\mathsf{SL}))^r). \end{aligned}$$

It then uses str to decrypt y and sends the answer to $\mathcal{A}'$.

At the end of the find stage, $\mathcal{A}'$ outputs two messages x_0 and x_1. Adversary $\mathcal{A}$ also outputs these two messages as the output of its find stage. Now $\mathcal{A}$ is given l encryptions (targets) $y_1, \ldots, y_l$ of x_b using the keys $K_1, \ldots, K_l$ respectively. Next, $\mathcal{A}$ randomly chooses $r_1, \ldots, r_l$ in Z_p. It forms l targets $T_1, \ldots, T_l$ by setting $T_j = (X_j = r_j P, y_j, \mathsf{SL}_j)$. These targets are given to $\mathcal{A}'$.

We now describe the simulation of the guess stage decryption queries of $\mathcal{A}'$ by $\mathcal{A}$. Let (X, y, SL) be a decryption query. There are several cases to consider:

Case 1: $X = X_j$ and $\mathsf{SL} = \mathsf{SL}_j$ for some $j \in \{1, \ldots, l\}$. We must have $y \neq y_j$ as otherwise $(X, y, \mathsf{SL}) = T_j$. $\mathcal{A}$ now queries its j-th decryption oracle $\mathcal{D}_{K_j}()$ on y_j and returns the answer to $\mathcal{A}'$.

Case 2: $(X, \mathsf{SL}) \neq (X_j, \mathsf{SL}_j)$ for any $j \in \{1, \ldots, l\}$. In this case, the query is dealt with as in the find stage.

Finally, $\mathcal{A}$ outputs whatever is produced by $\mathcal{A}'$. The above ensures a correct simulation by $\mathcal{A}$ of $\mathcal{A}'$ on $\mathcal{G}_l$. Also, the number of decryption queries made by $\mathcal{A}$ to all its decryption oracles is at most equal to q. Since $\mathcal{A}'$ runs in time t, the advantage of $\mathcal{A}'$ while running on $\mathcal{G}_l$ is

$$2|\mathsf{Pr}[\mathcal{A}'(\mathcal{G}_l) = 1] - 1/2| \leq \mathsf{Adv}^{\mathsf{sym}}(t, q).$$

This completes the proof. □

Now we are in a position to prove the main result.

Theorem 1.

$$\mathsf{Adv}^{\mathsf{HEAD}}(t, q) \leq \frac{2lp}{p - q} \times \mathsf{Adv}^{\mathsf{obdh}}(t, q) + \mathsf{Adv}^{\mathsf{sym}}(t, q).$$

Proof. Let $\mathcal{A}$ be any adversary for HEAD which runs in time at most t and makes at most q queries to the decryption oracle. Then from definition, we have

$$\mathsf{Adv}^{\mathsf{HEAD}}_{\mathcal{A}} = 2\,|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2|\,.$$

From Lemma 1, we have for each $i \in \{0, \ldots, l-1\}$,

$$|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1]| \leq \mathsf{Adv}^{\mathsf{obdh}}(t, q).$$

Now consider

$$\begin{aligned}
|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_0) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_l) = 1]| &= \left| \sum_{i=0}^{l-1} (\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1]) \right| \\
&\leq \sum_{i=0}^{l-1} |\mathsf{Pr}[\mathcal{A}(\mathcal{G}_i) = 1] - \mathsf{Pr}[\mathcal{A}(\mathcal{G}_{i+1}) = 1])| \\
&\leq l \times \frac{p}{p - q} \times \mathsf{Adv}^{\mathsf{obdh}}(t, q).
\end{aligned}$$

Further, from Lemma 2, we have $2|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_l) = 1] - 1/2| \leq \mathsf{Adv}^{\mathsf{sym}}(t, q)$. Let $\alpha = \mathsf{Pr}[\mathcal{A}(\mathcal{G}_0) = 1]$, $\beta = \mathsf{Pr}[\mathcal{A}(\mathcal{G}_l) = 1]$, $\gamma = (p/(p - q))\mathsf{Adv}^{\mathsf{obdh}}(t, q)$ and

$\delta = \mathsf{Adv}^{\mathsf{sym}}(t,q)$. Then we have $|\alpha - \beta| \leq l\gamma$ and $2|\beta - 1/2| \leq \delta$. The second inequality can be rewritten as $1/2 - \delta/2 \leq \beta \leq 1/2 + \delta/2$ and the first inequality can be rewritten as $\beta - l\gamma \leq \alpha \leq \beta + l\gamma$. Combining these two inequalitites, we obtain $1/2 - \delta/2 - l\gamma \leq \alpha \leq 1/2 + \delta/2 + l\gamma$ and hence $|\alpha - 1/2| \leq l\gamma + \delta/2$. Substituting the values of α, γ and δ gives us

$$2|\mathsf{Pr}[\mathcal{A}(\mathcal{G}_0) = 1] - 1/2| \leq \frac{2lp}{p-q} \times \mathsf{Adv}^{\mathsf{obdh}}(t,q) + \mathsf{Adv}^{\mathsf{sym}}(t,q).$$

Since $\mathcal{A}$ was chosen to be an arbitrary adversary (running in time at most t and making at most q queries to the decryption oracle), the above inequality holds for all such adversaries $\mathcal{A}$ and hence for any adversary which maximizes the advantage. This gives us

$$\mathsf{Adv}^{\mathsf{HEAD}}(t,q) \leq \frac{2lp)}{p-q} \times \mathsf{Adv}^{\mathsf{obdh}}(t,q) + \mathsf{Adv}^{\mathsf{sym}}(t,q).$$

This completes the proof. □

5 Concluding Remarks

In this paper, we have considered in detail the functionality of delegated decryption for encryption schemes. We have described an appropriate security model for such schemes. Our main contribution has been to present HEAD, which is an efficient and secure public key encryption scheme with delegated decryption capability.

References

1. http://www.i2r.a-star.edu.sg/icsd/staff/guilin/bible/proxy.htm.
2. M. Abdalla, M. Bellare and P. Rogaway. DHIES : An encryption scheme based on the Diffie-Hellman problem, *Proceedings of CT-RSA 2001*, Lecture Notes in Computer Science, Springer-Verlag, pages 143–158.
3. American National Standards Institute (ANSI) X9.F1 sub-committee. ANSI X9.63 Public Key Cryptography for the Financial Services Industry: Elliptic Curve Key Agreement and Transport Schemes, 1998. Working draft version 2.0.
4. D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. *Proceedings of CRYPTO 2001*, Lecture Notes in Computer Science, volume 2139, Springer-Verlag, pages 213–229.
5. D. Boneh, G. Di. Crescenzo, R. Ostrovsky and G. Persiano. Public Key Encryption with Keyword Search. *Proceedings of EUROCRYPT 2004*, Lecture Notes in Computer Science, Springer-Verlag, pp 506-522.
6. P. S. L. M. Barreto, H. Y. Kim and M. Scott. Efficient algorithms for pairing-based cryptosystems. *Proceedings of Crypto 2002*, Lecture Notes in Computer Science, volume 2442, Springer-Verlag, pages 354–368.
7. S. Galbraith, K. Harrison and D. Soldera. Implementing the Tate Pairing. *Proceedings of Algorithm Number Theory Symposium - ANTS V, 2002*, Lecture Notes in Computer Science, volume 2369, Springer-Verlag, pages 324–337.

8. S. Halevi and P. Rogaway. A tweakable enciphering mode. *Proceedings of Crypto 2003*, Lecture Notes in Computer Science, volume 2729, Springer-Verlag, pages 482–499.
9. S. Halevi and P. Rogaway. A parallelizable enciphering mode. *Proceedings of CT-RSA 2004*, Lecture Notes in Computer Science, Springer-Verlag, pages 292–304.
10. K. Kurosawa and T. Matsuo. How to remove MAC from DHIES. *Proceedings of ACISP, 2004*, Lecture Notes in Computer Science, Springer-Verlag, pp 236-247.
11. M. Mambo, K. Usuda and E. Okamoto. Proxy Signatures for Delegating Signing Operation. *Proceedings of the ACM Conference on Computer and Communications Security 1996*, pp 48-57.
12. Y. Mu, V. Varadharajan and K.Q. Nguyen. Delegated Decryption. *Proceeding of IMA Conference on Coding and Cryptography, 1999*, Lecture Notes in Computer Science, Springer-Verlag, Volume 1746, 258–269, 1999.
13. F. Zhang, R. Safavi-Naini and W. Susilo. An Efficient Signature Scheme from Bilinear Pairings and Its Applications. *Proceedings of Public Key Cryptography 2004*, Lecture Notes in Computer Science, volume 2947 , pp. 277-290. Springer-Verlag, 2004.

A Provably Secure Elliptic Curve Scheme with Fast Encryption

David Galindo[1], Sebastià Martín[1], Tsuyoshi Takagi[2], and Jorge L. Villar[1]

[1] Dep. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya, Campus Nord, c/Jordi Girona, 1-3, 08034 Barcelona
{dgalindo, sebasm, jvillar}@mat.upc.es
[2] Technische Universität Darmstadt, Fachbereich Informatik, Alexanderstr.10, D-64283 Darmstadt, Germany
ttakagi@cdc.informatik.tu-darmstadt.de

Abstract. We present a new elliptic curve cryptosystem with fast encryption and key generation, which is provably secure against passive adversaries in the standard model. The scheme uses arithmetic modulo n^2, where n is an RSA modulus, and merges ideas from Paillier and Rabin related schemes. Despite the typical bit length of n, our encryption algorithm is the fastest elliptic curve based encryption algorithm to the best of our knowledge, even faster than El Gamal elliptic curve encryption. The one-wayness (OW-CPA) of the new cryptosystem is as hard as factoring n while the semantic security (IND-CPA) is proved under a reasonable decisional assumption.

Two new length-preserving trapdoor permutations equivalent to factoring are also described.

Keywords: public-key cryptography, provable security, elliptic curves, fast encryption, Rabin-Paillier scheme.

1 Introduction

Several elliptic curve based cryptosystems have been proposed during the last decades. On the one hand, cryptosystems related to the elliptic curve discrete logarithm problem (such as elliptic curve versions of El Gamal) have the feature of having small keysizes, at the cost of moderate encryption/decryption times. On the other hand, cryptosystems based on elliptic curves over the ring $\mathbb{Z}_n$ have security related to the hardness of factoring $n = pq$. Therefore, their keysizes are the same as in RSA schemes while encryption/decryption times are greater. In both cases, messages are hidden by means of computing multiples of points. Thus, the computational cost depends on the size of the multiplier.

In this paper, a minimal encryption-time cryptosystem based on elliptic curves is proposed. The motivation comes from the Rabin-Paillier probabilistic encryption scheme [6]. Roughly speaking, that scheme works as follows. Let (n, e) be an RSA public key, $\mathbb{Z}_n$ be the ring of integers modulo n and Q_n be the set of squares in $\mathbb{Z}_n$. To encrypt a message $m \in \mathbb{Z}_n$, a random $r \in Q_n$ is

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 245–259, 2004.

used and the ciphertext is $r^{2e} + mn \bmod n^2$. The scheme is proved to be one-way under the factoring assumption and provides semantic security against passive adversaries under a reasonable decisional assumption.

The optimal encryption efficiency in the Rabin-Paillier scheme would be obtained using $e = 1$. In this case, the scheme is one-way but it is not semantically secure anymore, since the related decisional assumption is trivially solved. What we do is to design its analogue using elliptic curves over rings, and prove that in this case *semantic security is obtained* under a reasonable decisional assumption. As a result, the encryption algorithm is faster than those obtained in the elliptic curve versions of El Gamal. Furthermore, if the encryption efficiency is measured in terms of encryption time per plaintext bit, the difference is even greater.

As done in [11], the new cryptosystem works on a family of supersingular elliptic curves. Since doubling points on elliptic curves over $\mathbb{Z}_n$ is not a bijection, the set of allowed points must be restricted to the subset D_n of doubles of points. We show that if $p \equiv q \equiv 5 \bmod 12$ then doubling points in D_n is a trapdoor permutation whose one-wayness is equivalent to factoring n.

Now, by following the ideas in [3, 6], this bijection is lifted to $\mathbb{Z}_{n^2}$ and the definition of the new cryptosystem arises. Its semantic security is proven equivalent to deciding the existence of small roots of some polynomials. Since the best result in this area (namely [4]) does not apply to our case, the new problem is supposed to be intractable. To prove the scheme one-way, some interesting techniques are developed.

The rest of the paper is organised as follows. Section 2 is devoted to introduce the definition and some results about elliptic curves. In Section 3 we propose new trapdoor permutations equivalent to factoring. In Section 4, we describe the new scheme and prove that its one-wayness is based on the hardness of factoring the modulus. We also prove that the proposed scheme is semantically secure (IND-CPA) under a new assumption. Then, we argue why one should be confident on this new assumption. Finally, the computational cost of the new scheme is discussed in Section 5.

2 Some Results About Elliptic Curves

In this section, we are going to summarize the definition and some results about elliptic curves defined over the finite field $\mathbb{Z}_p$, and over the rings $\mathbb{Z}_{p^2}$ and $\mathbb{Z}_{n^2}$, where n is an RSA modulus.

Definition 1. *Let $p > 3$ be a prime. An elliptic curve over the finite field $\mathbb{Z}_p$, denoted by $E_p(a, b)$, where $a, b \in \mathbb{Z}_p$, and $\gcd(4a^3 + 27b^2, p) = 1$, is the set of points $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$ such that $y^2 = x^3 + ax + b \bmod p$, with a point $\mathcal{O}$ called* the point at infinity.

The set $E_p(a, b)$ is a group, with the usual tangent-and-chord operation. We will denote by $|E_p(a, b)|$ the number of elements of the group $E_p(a, b)$ and by $r\#P$ the r-th multiple of a point $P \in E_p(a, b)$. For an extensive treatment on elliptic curves we refer to [14], and for an overview on elliptic curve cryptosystems, see [13].

Elliptic curves can also be defined on the projective plane $\mathbb{P}^2(\mathbb{Z}_p)$ as the set of points $(x : y : z)$ satisfying $y^2z = x^3 + axz^2 + bz^3 \bmod p$, and $\gcd(x, y, z, p) = 1$. In particular, the point $(0 : 1 : 0)$ corresponds to the point at infinity $\mathcal{O}$. Following [5], this definition can be extended to the ring $\mathbb{Z}_{p^2}$. The natural map $\pi_p : E_{p^2}(a, b) \to E_p(a, b)$ that reduces coordinates modulo p, is a surjective group morphism whose kernel is the set $\{O_m = (mp : 1 : 0) \mid m \in \mathbb{Z}_p\}$, called the set of points at infinity.

$E_{n^2}(a, b)$ can be defined from the natural surjective maps from $E_{n^2}(a, b)$ to $E_{p^2}(a, b)$ and $E_{q^2}(a, b)$. Via the Chinese Remainder Theorem, $E_{n^2}(a, b)$ can be seen as a group isomorphic to $E_{p^2}(a, b) \times E_{q^2}(a, b)$. The natural group morphism from $E_{n^2}(a, b)$ to $E_n(a, b)$ will be denoted as π_n. These morphisms are depicted in Diagram 1.

$$\begin{array}{ccc} E_{n^2}(a,b) & \xrightarrow[\pi_{p^2} \times \pi_{q^2}]{\sim} & E_{p^2}(a,b) \times E_{q^2}(a,b) \\ \Big\downarrow{\scriptstyle \pi_n} & & \Big\downarrow{\scriptstyle \pi_p \times \pi_q} \\ E_n(a,b) & \xrightarrow[\pi_p \times \pi_q]{\sim} & E_p(a,b) \times E_q(a,b) \end{array}$$

Diagram 1: Some morphisms related to $E_{n^2}(a, b)$

Points on curves $E_{n^2}(a, b)$ can be classified in three types:

- Points at infinity: $O_m = (mn : 1 : 0)$, $m \in \mathbb{Z}_n$, (the kernel of π_n)
- Affine points: $(x, y) = (x : y : 1) \in E_{n^2}(a, b)$.
- Semi-infinite points: $(x : y : z) \in E_{n^2}(a, b)$, with $\gcd(z, n) = p$ or q.

The usual tangent-and-chord formulas allow to perform addition of affine points on $E_{n^2}(a, b)$, without knowledge of the factorisation of n. In particular, the formula to double an affine point is the following:

$$\boxed{2\#(x, y) = (\lambda^2 - 2x, -\lambda^3 + 3x\lambda - y), \text{ where } \lambda = (3x^2 + a)(2y)^{-1}.}$$

To deal with points at infinity the following addition formulas are used:

$$\boxed{\begin{array}{l} O_m + O_{m'} = O_{m+m'}. \\ (x, y) + O_m = (x - 2ymn, y - (3x^2 + a)mn). \end{array}}$$

3 New Trapdoor Permutations

In this section, the well-known Blum-Williams trapdoor permutation is adapted to the elliptic curve setting.

3.1 Blum-Williams Function

Let $n = pq$ be an RSA modulus with $p \equiv q \equiv 3 \bmod 4$, and let Q_n be the set of quadratic residues modulo n. The squaring function restricted to Q_n, i.e.

$$\begin{aligned} \mathcal{G}_n : Q_n &\longrightarrow Q_n \\ x &\longmapsto x^2 \bmod n \end{aligned}$$

is a trapdoor one-way permutation if factoring large numbers is unfeasible (see page 34 in [8]). Let us briefly recall how to invert $\mathcal{G}_n$, provided the factorisation of n (see [15] for a nice account on this). We first compute the numbers $f = c^{\frac{p+1}{4}} \bmod p$ and $g = c^{\frac{q+1}{4}} \bmod q$, which are the square roots of c modulo p and modulo q that are quadratic residues to their respective modulus. Then, by using the Chinese Remainder Theorem, we obtain an $s \in Q_n$ such that $s^2 = c \bmod n$.

3.2 Point-Doubling Trapdoor Permutation

As in KMOV scheme [11], only supersingular curves $E_n(0, b)$, $b \in \mathbb{Z}_n^*$, will be considered; in particular curves with $p \equiv q \equiv 2 \bmod 3$. A new restriction on the prime factors of n must be introduced, in order to avoid the existence of points of order 4.

Observation 1. *If $p \equiv 5 \bmod 12$, then $|E_p(0, b)| \equiv 2 \bmod 4$, and consequently there are no points of order 4 on $E_p(0, b)$. Also, there is a unique point of order 2, namely $(\eta, 0)$, where η is the unique cubic root of $-b$. This implies that given a point $P \in E_p(0, b)$, the equation $2\#\bar{P} = 2\#P$ has exactly two solutions: $\bar{P} = P$ and $\bar{P} = P + (\eta, 0)$, since the order of the point $\bar{P} - P$ divides 2.*

Now, the elliptic analogue to the set of quadratic residues is defined.

Definition 2. *For $n = pq$, and $p \equiv q \equiv 5 \bmod 12$, let*

$$D_n = \{2\#(x, y) \in \mathbb{Z}_n \times \mathbb{Z}_n \mid x \in \mathbb{Z}_n,\ y \in \mathbb{Z}_n^*,\ y^2 - x^3 \in \mathbb{Z}_n^*\},$$

where the double $2\#(x, y)$ is computed on the curve $E_n(0, b)$, with $b = y^2 - x^3$.

We say that $(x, y) \in \mathbb{Z}_n \times \mathbb{Z}_n^*$ is a *double* if it is in D_n. We will also consider the sets D_p and D_q defined in the same way as D_n, but using modulo p and q instead of n. From the Chinese Remainder Theorem, it is clear that $D_n = D_p \times D_q$.

Lemma 1. *If $(u, v) \in D_n$, then $v \in \mathbb{Z}_n^*$.*

Proof. Let $Q = (u, v) \in D_n$. Then, there exists a point $P = (x, y)$ on the same curve such that $Q = 2\#P$ and $y \in \mathbb{Z}_n^*$. Let us suppose that $v = 0 \bmod p$. This implies that $2\#\pi_p(Q) = O$ and then $4\#\pi_p(P) = O$. Since there are no points of order 4 on $E_p(0, b)$, we can assure that $2\#\pi_p(P) = O$. So, $y \equiv 0 \bmod p$, which is a contradiction. □

Lemma 2. $|D_p| = \frac{(p-1)^2}{2}$ *and* $|D_n| = \frac{(p-1)^2(q-1)^2}{4}$.

Proof. Let $Q \in E_p(0, b) \cap D_p$ where $b \in \mathbb{Z}_p^*$. From observation 1 it is clear that the equation $2\#P = Q$ has exactly two solutions $P, \bar{P} \in E_p(0, b)$. Since there are $p-1$ affine points $P = (x, y)$ on $E_p(0, b)$ with $y \in \mathbb{Z}_p^*$, then $|E_p(0, b) \cap D_p| = \frac{p-1}{2}$. By considering the $p - 1$ possible values for b, we obtain the claimed result $|D_p| = \frac{(p-1)^2}{2}$. Finally, $|D_n| = \frac{(p-1)^2(q-1)^2}{4}$ comes from $D_n = D_p \times D_q$. □

Proposition 1. *Let $n = pq$, with $p \equiv q \equiv 5 \bmod 12$. Then, the following map is a bijection:*

$$\begin{aligned} \Delta_n : D_n &\longrightarrow D_n \\ (x, y) &\longmapsto 2\#(x, y) \end{aligned}$$

Proof. Δ_n is well-defined by the definition of D_n and lemma 1. In order to prove that Δ_n is injective, let us consider Q_1 and Q_2 in D_n such that $2\#Q_1 = 2\#Q_2$. This implies, on the one hand, that there exist P_1 and P_2 such that $Q_1 = 2\#P_1$ and $Q_2 = 2\#P_2$. On the other hand, P_1, P_2, Q_1 and Q_2 lie on the same curve and $2\#(Q_2 - Q_1) = O$. Thus, $4\#(P_2 - P_1) = O$ which implies $2\#P_2 = 2\#P_1$, since there are no points of order 4 in $E_n(0, b)$. Therefore, $Q_2 = Q_1$. Finally, by a simple counting argument, Δ_n must be surjective. □

We point out that Δ_n is an elliptic analogue of Blum-Williams function.

Proposition 2. *If $p \equiv q \equiv 5 \bmod 12$, then Δ_n is a trapdoor permutation equivalent to factoring n.*

Proof. Let us see, given the trapdoor information, p and q, how to invert Δ_n efficiently on a point $Q \in D_n$. Since Δ_n is a bijection, there exist a point $P \in D_n$ such that $Q = 2\#P$, but there also exists another point $R \in D_n$ such that $P = 2\#R$, that is $Q = 4\#R$. Let us consider the points $T_p = \frac{p+3}{4}\#\pi_p(Q)$ and $T_q = \frac{q+3}{4}\#\pi_q(Q)$. Then, $T_p = (p+3)\#\pi_p(R) = 2\#\pi_p(R) = \pi_p(P)$ and $T_q = (q+3)\#\pi_q(R) = 2\#\pi_q(R) = \pi_q(P)$. Thus, the preimage P of Q can be easily computed from T_p and T_q by the Chinese Reminder Theorem. In fact, a point-halving procedure that works in a more general case can be found in [11].

Now, to conclude the proof, it suffices to show a reduction from the one-wayness of Δ_n to the problem of factoring n. To do this, take a random pair $\bar{P} = (\bar{x}, \bar{y}) \in \mathbb{Z}_n \times \mathbb{Z}_n^*$ and compute $Q = 2\#\bar{P}$, that is uniformly distributed in D_n. Observe that $\pi_q(\bar{P}) \in D_q$ but $\pi_p(\bar{P}) \notin D_p$ with probability $1/4$. Let us consider we are in this case. Since $Q \in D_n$, there exists a point $P = (x, y) \in D_n$ such that $Q = 2\#P$. Let us consider an algorithm $\mathcal{A}$ such that on input (n, Q) returns P with probability ϵ. If $\mathcal{A}$ succeeds then $2\#\bar{P} = 2\#P$. We can assure now that $\pi_q(\bar{P}) = \pi_q(P)$ and $\bar{x} \neq x \bmod p$ (note that, if $\bar{x} = x \bmod p$, then $\pi_p(\bar{P}) = \pm\pi_p(P)$ and $\pi_p(\bar{P}) \in D_p$, which is a contradiction). Finally, $\gcd(\bar{x} - x, n) = p$. By considering also the case $\pi_p(\bar{P}) \in D_p$ and $\pi_q(\bar{P}) \notin D_q$, it is straightforward to show that this procedure gives a nontrivial factor of n with probability $\epsilon/2$. □

3.3 Lifted Trapdoor Bijection

Next, a lifted version of the map Δ_n is presented. The technique used here is somewhat related to the one used in [6]. The following useful property allows to lift a point $P_0 \in E_n(0, b_0)$ to a special point P on each curve $E_{n^2}(0, b)$ such that $b \equiv b_0 \bmod n$.

Property 1. Let $b \in \mathbb{Z}_{n^2}^*$ and $P = (x_0, y_0) \in E_n(0, b \bmod n)$, with $y_0 \in \mathbb{Z}_n^*$. Then, there exists a unique point $(x_0, y) \in E_{n^2}(0, b)$ such that $y \equiv y_0 \bmod n$.

Proof. Let $y = y_0 + \gamma n \in \mathbb{Z}_{n^2}^*$, where $\gamma \in \mathbb{Z}_n$. Then, (x_0, y) belongs to $E_{n^2}(0, b)$ if and only if

$$\gamma = \frac{x_0^3 - y_0^2 + b}{n}(2y_0)^{-1} \bmod n.$$

□

Let $n = pq$, with $p \equiv q \equiv 5 \bmod 12$, and let us consider the following sets:

$$\Omega_n = \{(x, y) \in \mathbb{Z}_{n^2} \times \mathbb{Z}_{n^2}^* \mid \pi_n(x, y) \in D_n\},$$

$$\omega_n = \{(x, y) \in \Omega_n \mid x < n\}.$$

and the function

$$\begin{aligned} \psi_n : \omega_n \times \mathbb{Z}_n &\longrightarrow \Omega_n \\ (x, y, m) &\longrightarrow 2\#P + O_m \end{aligned}$$

where $P = (x, y)$, and the double as well as the addition are performed on $E_{n^2}(0, b)$, with $b = y^2 - x^3 \bmod n^2$.

Lemma 3. *If $p \equiv q \equiv 5 \bmod 12$, then the map ψ_n is well defined and bijective.*

Proof. The map ψ_n is well-defined since $\psi_n(x, y, m)$ is always in Ω_n. This comes from the definition of Ω_n, since $\psi_n(x, y, m) \in \Omega_n$ if and only if $\pi_n(\psi_n(x, y, m)) \in D_n$. As $(x, y) \in \omega_m$, $\pi_n(x, y) \in D_n$ and then $\pi_n(\psi_n(x, y, m)) = \pi_n(2\#(x, y)) = 2\#\pi_n(x, y) \in D_n$.

In order to show that ψ_n is injective, let us suppose $\psi_n(x, y, m) = \psi_n(x', y', m')$ for some $(x, y), (x', y') \in \omega_n$ and $m, m' \in \mathbb{Z}_n$. Reducing this equality modulo n, we obtain $2\#\pi_n(x, y) = 2\#\pi_n(x', y')$. By the injectivity of Δ_n and from the fact that $\pi_n(x, y)$ and $\pi_n(x, y)$ are points in D_n we deduce $\pi_n(x, y) = \pi_n(x', y')$.

Now, taking into account that (x, y), (x', y') belong to the same curve $E_{n^2}(0, b)$, and that $0 \leq x, x' < n$, we use Property 1 to deduce $(x, y) = (x', y')$. From this, it is easy to see that $O_m = O_{m'}$, so $m = m'$.

Finally, let us show that ψ_n is surjective. Let $C = (u, v) \in \Omega_n$ and $b = v^2 - u^3 \bmod n^2$. Then there exists $P_0 = (x_0, y_0) \in D_n$ such that $\pi_n(u, v) = 2\#P_0$. Let $P = (x_0, y)$ be the point on $E_{n^2}(0, b)$ given in Property 1. Clearly, $P \in \omega_n$ and $2\#P - C$ is a point at infinity, say O_m. Then, $C = \psi_n(x_0, y, m)$. □

Proposition 3. *If $p \equiv q \equiv 5 \bmod 12$, then ψ_n is a trapdoor bijection equivalent to factoring n.*

Proof. Let us see, given the trapdoor information, p and q, how to invert ψ_n efficiently on a point $C = (u, v) \in \Omega_n$. Let $b = v^2 - u^3 \bmod n^2$. Compute $Q_0 = \pi_n(C)$ that is a point in D_n and let $P_0 \in D_n$ such that $Q_0 = 2\#P_0$. The point P_0 can be efficiently computed by using the procedure for inverting Δ_n described in the proof of proposition 2. Then, let $P = (x, y) \in E_{n^2}(0, b)$ the point given in property 1 computed from P_0. Clearly, $P \in \omega_n$ and $C - 2\#P$ is a point at infinity, say O_m. Then, $C = \psi_n(x, y, m)$.

To conclude the proof, it suffices to show a reduction from the one-wayness of ψ_n to the problem of factoring n. As in the proof of proposition 2, take a random pair $(\bar{x}, \bar{y}) \in \mathbb{Z}_n \times \mathbb{Z}_n^*$ and compute $Q_0 = (u_0, v_0) = 2\#(\bar{x}, \bar{y})$. Now randomly lift Q_0 obtaining $C = (u_0 + \mu n, v_0 + \nu n)$, where μ and ν are randomly selected in $\mathbb{Z}_n$. Note that C is uniformly distributed on Ω_n. Let us consider an algorithm $\mathcal{A}$ such that on input (n, C) returns $P = (x, y) \in \omega_n$ and $m \in \mathbb{Z}_n$ such that $C = \psi_n(x, y, m)$, with probability ϵ. If $\mathcal{A}$ succeeds, then $\Delta_n(\pi_n(x, y)) = 2\#\pi_n(x, y) = \pi_n(C) = Q_0$. So, by following the same steps as in the proof of proposition 2, a nontrivial factor of n is found with probability $\epsilon/2$. □

4 The New Scheme

Based on the previous trapdoor bijection, in this section we present an elliptic curve cryptosystem (ECC) over the ring $\mathbb{Z}_{n^2}$ which is semantically secure against passive adversaries under a new decisional assumption a nd has the fastest encryption and the highest one-way security among the known ECC, in the standard model.

Key generation. Given a security parameter ℓ, choose at random two primes p and q with ℓ bits such that $p \equiv q \equiv 5 \bmod 12$. Then the public key is PK=$\{n\}$, $n = pq$, and the private key is SK=$\{p, q\}$.

Encryption. To encrypt a message $m \in \mathbb{Z}_n$ we choose at random $z \in \mathbb{Z}_n$ and $t \in \mathbb{Z}_n^*$ such that $b_0 = t^2 - z^3 \in \mathbb{Z}_n^*$. This choice determines an elliptic curve $E_n(0, b_0)$ and a point $R = (z, t)$ on it. Let $P_0 = (x_0, y_0) = 2\#R$ and γ chosen at random in $\mathbb{Z}_n$, and compute $y = y_0 + \gamma n$. Then $P = (x_0, y)$ is a random point in ω_n. The encryption of the message $m \in \mathbb{Z}_n$ is $C = \psi_n(x_0, y, m)$.

Decryption. To recover the message m from the ciphertext $C = (u, v) = \psi_n(x, y, m)$, the randomness $(x, y) \in \omega_n$ is computed firstly and, afterwards, m is easily obtained from $O_m = C - 2\#(x, y)$. This is just the procedure detailed in the proofs of propositions 2 and 3. We recall the steps to obtain (x, y) from C. Firstly, compute $\pi_n(x, y)$ by inverting Δ_n on $\pi_n(C)$ (using the Chinese Reminder Theorem). Next, compute $(x, y) \in E_{n^2}(0, b)$, where $b = v^2 - u^3 \bmod n^2$, by using property 1.

In the following, the security of this scheme is analyzed. Let us introduce some convenient notations. If A is a finite set, $x \leftarrow A$ will denote that x is randomly selected with uniform distribution in A. We will denote by $D_1 \approx D_2$ the fact that two probability distributions D_1 and D_2 are polynomially indistinguishable. Notice that if g is a bijection such that g and g^{-1} can be computed in probabilistic polynomial time, then $D_1 \approx D_2$ is equivalent to $g(D_1) \approx g(D_2)$.

M_ℓ will denote the set of integers $n = pq$ such that p and q are two primes with ℓ bits, and $p \equiv q \equiv 5 \bmod 12$.

4.1 One-Wayness

The following lemma allows to compute, with overwhelming probability, a rational function of the coordinates of a point $P_0 \in D_n$, given two special lifted points Q_1 and Q_2 such that $\pi_n(Q_1) = \pi_n(Q_2) = 2\#P_0$.

Lemma 4. *Let $Q_1 = (u_1, v_1) = 2\#P_1$ and $Q_2 = (u_2, v_2) = 2\#P_2$ where P_1 and P_2 are different points in ω_n such that $\pi_n(P_1) = \pi_n(P_2)$. Let $b_1 = v_1^2 - u_1^3 \bmod n^2$ and $b_2 = v_2^2 - u_2^3 \bmod n^2$. Let $(x_0, y_0) = \pi_n(P_1)$. Then*

$$9\alpha \left(\frac{x_0}{y_0}\right)^4 = -4\beta \bmod n,$$

where $\alpha = (b_2 - b_1)/n$ and $\beta = (u_2 - u_1)/n$.

Proof. Since $P_1, P_2 \in \omega_n$ we can write $P_1 = (x_0, y_1)$ and $P_2 = (x_0, y_2)$, where $y_1 \equiv y_2 \equiv y_0 \bmod n$ and $x_0 < n$. Observe that both points lie in different curves. Indeed, Q_1 and P_1 are in $E_n(0, b_1)$ while Q_2 and P_2 are in $E_n(0, b_2)$. Since $b_1 \equiv b_2 \bmod n$, $\alpha = (b_2 - b_1)/n$ is well defined.

By using the doubling formula, we obtain

$$u_1 = \left(\frac{3x_0^2}{2y_1}\right)^2 - 2x_0 = \frac{9x_0^4}{4(x_0^3 + b_1)} - 2x_0 \bmod n^2$$
$$u_2 = \left(\frac{3x_0^2}{2y_2}\right)^2 - 2x_0 = \frac{9x_0^4}{4(x_0^3 + b_2)} - 2x_0 \bmod n^2$$

and then,

$$u_2 - u_1 = \frac{9x_0^4}{4(x_0^3 + b_2)} - \frac{9x_0^4}{4(x_0^3 + b_1)} = \frac{9x_0^4(b_1 - b_2)}{4(x_0^3 + b_1)(x_0^3 + b_2)} = -\frac{9}{4}\frac{x_0^4}{y_1^2 y_2^2}\alpha n \bmod n^2.$$

Therefore

$$\beta = \frac{u_2 - u_1}{n} = -\frac{9}{4}\left(\frac{x_0}{y_0}\right)^4 \alpha \bmod n.$$

□

Note that if Q_1 and Q_2 are chosen at random (but fulfilling the conditions in lemma 4) then $\alpha \in \mathbb{Z}_n^*$ with overwhelming probability.

From this lemma, given a random modulus n, we can exploit an adversary $\mathcal{A}$ against the one-wayness of the proposed scheme to build such two points Q_1 and Q_2, and efficiently derive a nontrivial factor of n.

Proposition 4. *The one-wayness* (OW-CPA) *of the proposed scheme is equivalent to the unfeasability of factoring the modulus.*

Proof. Let $\mathcal{A}$ be an adversary trying to break the one-wayness of the proposed cryptosystem. Let us consider the following probability

$$\mathsf{Succ}_{\mathcal{A}}^{\mathsf{OW}}(\ell) = \mathsf{Prob}\left[\mathcal{A}(n, \psi_n(x, y, m)) = m \mid n \leftarrow M_\ell;\ (x, y) \leftarrow \omega_n;\ m \leftarrow \mathbb{Z}_n\right].$$

The following algorithm $\mathcal{B}$ can be used to obtain a nontrivial factor of $n \leftarrow M_\ell$.

$\mathcal{B}(n)$

1 $\bar{x}_0 \leftarrow \mathbb{Z}_n;\ \bar{y}_0 \leftarrow \mathbb{Z}_n;\ b_0 = \bar{y}_0^2 - \bar{x}_0^3 \bmod n$
2 if $\gcd(\bar{y}_0, n) \neq 1$ return $\gcd(\bar{y}_0, n)$
3 if $\gcd(b_0, n) \neq 1$ return $\gcd(b_0, n)$
4 $(u_0, v_0) = 2\#(\bar{x}_0, \bar{y}_0)$, computed in $E_n(0, b_0)$
5 $\gamma_1 \leftarrow \mathbb{Z}_n;\ \delta_1 \leftarrow \mathbb{Z}_n;\ C_1 = (u_0 + \gamma_1 n, v_0 + \delta_1 n)$
6 $m_1 = \mathcal{A}(n, C_1);\ (u_1, v_1) = C_1 - O_{m_1}$
7 $\gamma_2 \leftarrow \mathbb{Z}_n;\ \delta_2 \leftarrow \mathbb{Z}_n;\ C_2 = (u_0 + \gamma_2 n, v_0 + \delta_2 n)$
8 $m_2 = \mathcal{A}(n, C_2);\ (u_2, v_2) = C_2 - O_{m_2}$
9 $\alpha = (v_2^2 - u_2^3 - v_1^2 + u_1^3)/n$
10 if $\gcd(\alpha, n) \neq 1$ return $\gcd(\alpha, n)$
11 $\beta = (u_2 - u_1)/n$
12 return $\gcd\left(\frac{\bar{x}_0^4}{\bar{y}_0^4} + \frac{4\beta}{9\alpha}, n\right)$

At steps 1 to 4 of the algorithm, a random point $Q_0 = (u_0, v_0) \in D_n$ is built. Next, points $Q_1 = (u_1, v_1)$ and $Q_2 = (u_2, v_2)$ are built by calling $\mathcal{A}$ twice using two randomly lifted points C_1 and C_2 coming from the same point Q_0.

If $\mathcal{A}$ succeeds in the first call, at step 6, then Q_1 can be written as $Q_1 = 2\#P_1$ where $P_1 \in \omega_n$. This is a consequence of the bijectivity of ψ_n, since $C_1 \in \Omega_n$, and then there exists a unique $P_1 \in \omega_n$ and a unique $m_1 \in \mathbb{Z}_n$ such that $C_1 = \psi_n(P_1, m_1)$. The same occurs with $Q_2 = 2\#P_2$, if $\mathcal{A}$ succeeds in the second call.

Let us consider the case that $\mathcal{A}$ succeeds in both calls. Note that $Q_0 = \pi_n(C_1) = \pi_n(C_2)$ and $Q_0 = 2\#\pi_n(P_1) = 2\#\pi_n(P_2)$. But there is only one point in D_n whose double is Q_0. Thus, $\pi_n(P_1) = \pi_n(P_2)$. Let $P_0 = (x_0, y_0) = \pi_n(P_1) = \pi_n(P_2)$. Since Q_1 and Q_2 fulfil the conditions in the previous lemma, then

$$\left(\frac{x_0}{y_0}\right)^4 = -\frac{4\beta}{9\alpha} \bmod n$$

if $\alpha \in \mathbb{Z}_n^*$.

On the other hand, $Q_0 = 2\#(\bar{x}_0, \bar{y}_0) = 2\#P_0$. Observe that $P_0 \in D_n$ but $\bar{P}_0 = (\bar{x}_0, \bar{y}_0)$ is chosen at random. By using the Chinese Reminder Theorem, $\pi_p(\bar{P}_0) = \pi_p(P_0)$ with probability 1/2, and independently $\pi_q(\bar{P}_0) = \pi_q(P_0)$ with probability 1/2. So, with probability 1/4, $\pi_q(\bar{P}_0) = \pi_q(P_0)$ but $\pi_p(\bar{P}_0) \neq \pi_p(P_0)$. The last expression implies $\bar{x}_0 \neq x_0 \bmod p$. To see this, let us suppose $\bar{x}_0 = x_0 \bmod p$. Then, $\pi_p(\bar{P}_0) = -\pi_p(P_0)$. From $2\#\bar{P}_0 = 2\#P_0$ we deduce $4\#\pi_p(\bar{P}_0) = O$. Since there are no points with order 4 on $E_p(0, b_0 \bmod p)$ then $2\#\pi_p(\bar{P}_0) = O$ and consequently $\bar{y}_0 \equiv 0 \bmod p$. But, this is not possible due to step 2 in the algorithm.

Except for a negligible fraction of the values of $(\bar{x}_0, \bar{y}_0)$, it can be also shown that[1]

[1] The exception are points $(\bar{x}_0, \bar{y}_0)$ such that $\bar{x}_0 \bmod p$ is a root of a certain polynomial of degree 8. However, by making some cumbersome calculations, it can be shown that if $p \equiv 1 \bmod 8$ then there are no exceptional points, otherwise, i.e. $p \equiv 5 \bmod 8$, there are only $p - 1$ exceptional points (modulo p), that is, only a fraction $1/p$. (See appendix B for details.)

$$\left(\frac{\bar{x}_0}{\bar{y}_0}\right)^4 \neq \left(\frac{x_0}{y_0}\right)^4 \bmod p.$$

Then, by using lemma 4,

$$\gcd\left(\frac{\bar{x}_0^4}{\bar{y}_0^4} + \frac{4\beta}{9\alpha}, n\right) = p.$$

By considering the other case, $\pi_p(\bar{P}_0) = \pi_p(P_0)$ but $\pi_q(\bar{P}_0) \neq \pi_q(P_0)$, the previous gcd expression leads to the other nontrivial factor of n.

Finally, except for a negligible function of ℓ (due to the technical steps 2, 3 and 10, and the anomalous values of $(\bar{x}_0, \bar{y}_0)$) the success probability

$$\mathsf{Succ}_{\mathcal{B}}^{\mathsf{FACT}}(\ell) = \mathsf{Prob}\left[\mathcal{B}(n) \in \{p, q\} \mid n \leftarrow M_\ell\right]$$

is one half the probability that $\mathcal{A}$ is successful in both calls. Notice that these two calls are not independent, since they share the same values of n and Q_0. However, by using lemma 5 (given in appendix A) with algorithm $\mathcal{A}$, predicate $P =$ "$\mathcal{A}$ succeeds" and map $f(n, C) = (n, \pi_n(C))$, the following inequality is obtained:

$$\mathsf{Succ}_{\mathcal{B}}^{\mathsf{FACT}}(\ell) \geq \frac{1}{2}\left(\mathsf{Succ}_{\mathcal{A}}^{\mathsf{OW}}(\ell)\right)^2.$$

□

4.2 Semantic Security

The scheme is semantically secure under the following assumption:

Assumption 1 (Decisional Small-x Double (DSD) Assumption). *The following probability distributions are polynomially indistinguishable*

$$\begin{aligned} D_{\text{double}} &= (n, 2\#(x, y)) \quad \text{where } n \leftarrow M_\ell,\ (x, y) \leftarrow \omega_n \\ D_{\text{random}} &= (n, (x', y')) \quad \text{where } n \leftarrow M_\ell,\ (x', y') \leftarrow \Omega_n. \end{aligned}$$

Proposition 5. *The proposed scheme is semantically secure* (IND-CPA) *if and only if the* DSD *assumption holds.*

Proof. Semantic security is equivalent to indistinguishability of encryptions, so we have to prove that for all $m_0 \in \mathbb{Z}_n$, the distributions

$$\begin{aligned} D_0 &= (n, \psi_n(x, y, m_0)) \quad \text{where } n \leftarrow M_\ell,\ (x, y) \leftarrow \omega_n\,, \quad \text{and} \\ D &= (n, \psi_n(x, y, m)) \quad \text{where } n \leftarrow M_\ell,\ (x, y) \leftarrow \omega_n,\ m \leftarrow \mathbb{Z}_n. \end{aligned}$$

are polynomially indistinguishable. From the definition of sum of an affine point and a point at infinity given at the end of section 2, it is easy to see that the map

$$\begin{aligned} \Omega_n &\longrightarrow \Omega_n \\ P &\longmapsto P - O_{m_0} \end{aligned}$$

is a polynomial time bijection. Then, $D_0 \approx D$ is equivalent to

$$(n, 2\#(x,y)) \approx (n, 2\#(x,y) + O_{m'}), \quad \text{with } (x,y) \leftarrow \omega_n,\ m' \leftarrow \mathbb{Z}_n\,.$$

Note that the distribution on the left side is D_{double}. Besides, since $2\#(x,y) + O_{m'} = \psi_n(x,y,m')$, and ψ_n is a bijection, then D and D_{random} are identical distributions. □

Finally, we argue why one should be confident about the hardness of the new decisional problem presented in this paper.

According to the formula for computing the double of a point on an elliptic curve $E_{n^2}(0,b)$ (see end of Section 2), given $(u,v) = 2\#(x_1,y_1)$, x_1 is a root of the univariate polynomial $R(x) = x^4 + 4x^3u - 8bx + 4bu \in \mathbb{Z}_{n^2}[x]$. Then, DSD assumption is related to the difficulty of deciding if the polynomial $R(x)$ has a root smaller than n.

Similarly, the semantic security of other related cryptosystems (such as [3]) is related to the difficulty of deciding if a certain polynomial has a root smaller than n. The best known way to attack the above decisional problems is to solve their computational versions. The problem of finding small roots of polynomials modulo a large integer with unknown factorisation has been directly studied in the literature. The most powerful result in this area was obtained by Coppersmith in [4]. This result ensures that one can efficiently compute (i.e. in polynomial time) all roots x_1 of a polynomial $P(x) \in \mathbb{Z}_K[x]$ with degree d such that $|x_1| < K^{1/d}$. Up to now, despite a lot of research in this area (for instance in [9, 1, 10]), no improvement on this bound has been made. The result by Coppersmith implies we can only find the roots $|x_1| < (n^2)^{1/4} = n^{1/2}$ of the polynomial $R(x)$, which does not affect the validity of DSD assumption.

5 Efficiency Analysis

Now we study the encryption cost of our scheme. Since operations modulo a large number are involved, we neglect the cost of performing additions, multiplications and divisions by small integers. We will express the cost in terms of multiplications $\bmod n$, because modular inverses can be computed within a constant number of modular multiplications.

Generating $(x,y) \in \omega_n$*:* 5 multiplications modulo n, 1 inverse modulo n, and 1 n-length integer multiplication.

Computing $2\#(x,y)$*:* 5 multiplications modulo n^2, 1 inverse modulo n^2.

Adding O_m*:* 3 multiplications modulo n, 2 n-length integer multiplication.

We point out that $a^{-1} \bmod n^2$ can be obtained by computing $a^{-1} \bmod n$ and then performing two multiplications modulo n^2. Let c be the number of multiplications modulo n needed to compute $a^{-1} \bmod n$. Since the cost of multiplying two numbers mod n^2 is roughly the cost of 4 multiplications modulo n, we deduce

that $a^{-1} \bmod n^2$ can be computed in $8 + c$ multiplications modulo n. Practical implementations, suggests than the value $c = 8$ can be taken (see [2]).

Then, since the n-length integer multiplication cost is bounded by the cost of a multiplication modulo n, the encryption cost of our scheme is 55 multiplications modulo n. Thus we have proved that our scheme is drastically more efficient than the previous semantically secure elliptic curve cryptosystems (ECC) in the standard model based on factoring.

Next, we will compare the efficiency of our scheme with the well-known El Gamal ECC scheme. We assume that El Gamal ECC is performed over $\mathbb{Z}_p^*$, where p is 160 bits long, and our scheme is performed over $\mathbb{Z}_{n^2}^*$, where n is 1024 bits long (cf. [12]). We will express both encryption costs in terms of multiplications modulo n.

In El Gamal ECC the most time consuming operation is the computation of two multiples $r\#P$ and $ra\#P$, where r is a random integer whose size is roughly the same as the modulus p, and a is a fixed integer. Then, using the *double and add* algorithm, the computation of these two multiples requires on average k additions of points and $2k$ doublings, where k is the bit length of r. Assuming that a point addition or doubling requires about 12 modular multiplications, then El Gamal ECC would take approximately $3 \cdot 160 \cdot 12$ multiplications modulo p. Since the time needed to perform a modular multiplication is quadratic in the size of the modulus, the ratio between the time of a multiplication modulo p and a multiplication modulo n is $\frac{160^2}{1024^2}$. It follows that the encryption time of El Gamal ECC would be equivalent to 159 multiplications modulo n, which is almost three times the encryption cost of our scheme. If the efficiency is measured in terms of encryption-time per plaintext bit, this ratio must be multiplied by the ratio of the message lengths. Therefore, our cryptosystem is 18 times faster than El Gamal ECC in encryption-time per bit.

Thus, *our cryptosystem is the provably secure* IND-CPA *elliptic curve cryptosystem in the standard model with the fastest encryption algorithm* to the best of our knowledge. The key generation of the proposed cryptosystem is also very fast; indeed it is even faster than generating an RSA key, since only the modulus is needed. Regarding decryption, the main cost is due to the computation of $\frac{p+3}{4}\#P \in E_p(0, b)$, and $\frac{q+3}{4}\#P \in E_q(0, b)$, from $P \in E_n(0, b)$ which is almost the same as in the other existing ECC over $\mathbb{Z}_{n^2}$. Nevertheless, due to its decryption cost, it is unlikely that our scheme could compete with El Gamal ECC.

Acknowledgments

This work was partially supported by Sapanish Ministerio de Ciencia y Tecnologá under project TSC 2003-008660.

References

1. D. Boneh and G. Durfee. Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. *EUROCRYPT 1999* LNCS **1592** 1–11 (1999)

2. R. P. Brent. Some Integer Factorization Algorithms using Elliptic Curves. *Australian Computer Science Comunications* 24–26 (1986) (Republished 1998).
3. D. Catalano, R. Gennaro, N. Howgrave-Graham and P. Q. Nguyen. Paillier's Cryptosystem Revisited.*ACM CCS '2001 ACM Press* (2001).
4. D. Coppersmith. Finding a small root of a univariate modular equation. *EUROCRYPT '96, LNCS* **1070** 155–165 (1996).
5. S. Galbraith. Elliptic Curve Paillier Schemes. *Journal of Cryptology* 15 (2) 129–138 (2002).
6. D. Galindo, S. Martín, P. Morillo and J. L. Villar. A Practical Public Key Cryptosystem from Paillier and Rabin Schemes. *PKC'03* LNCS **2567** 279–291 (2002).
7. D. Galindo, S. Martín, P. Morillo and J. L. Villar. An efficient semantically secure elliptic curve cryptosystem based on KMOV. *Proceedings of International Workshop on Coding and Cryptography (WCC'03)*, (2003).
8. S. Goldwasser and M. Bellare. Lecture Notes on Cryptography. `http://www-cse.ucsd.edu/users/mihir`
9. N.A. Howgrave-Graham. Computational Mathematics Inspired by RSA. PhD Thesis. University of Bath (1999)
10. N.A. Howgrave-Graham. Approximate Integer Common Divisors. *CaLC'01, LNCS* **2146** 51–66 (2001)
11. K. Koyama, U.M. Maurer, T. Okamoto and S.A. Vanstone. New Public-Key Schemes Based on Elliptic Curves over the Ring $\mathbb{Z}_n$. *CRYPTO '91, LNCS* **576** 252–266 (1991).
12. A. K. Lenstra and E. R. Verheul. Selecting Cryptographic Key Sizes. http://cryptosavvy.com/cryptosizes.pdf
13. A. Menezes. Elliptic Curve Public-Key Cryptosystems. *Kluwer Academic SECS* **234** (1993)
14. J.H. Silverman. The arithmetic of elliptic curves. *Springer GTM* **106** (1986).
15. H.C.A. van Tilborg. A Professional Reference and Interactive Tutorial. *Kluwer Academic Publishers SECS* **528** (1999).

A Technical Lemma

This technical lemma is useful when dealing with two non-independent calls to a probabilistic algorithm.

Lemma 5. *Consider a probabilistic algorithm $\mathcal{A}$ with input $x \in X$, a (surjective) map $f : X \to Y$, and a predicate P on the input and the output of $\mathcal{A}$ (e.g. $P(n, \mathcal{A}(n))$ true if $\mathcal{A}(n)$ is a nontrivial factor of n).*

Let $\epsilon = \mathsf{Prob}\left[P(x, \mathcal{A}(x)) \mid x \leftarrow X\right]$. Then,

$$\mathsf{Prob}\left[P(x_1, \mathcal{A}(x_1)) \wedge P(x_2, \mathcal{A}(x_2)) \mid x_1 \leftarrow X;\ x_2 \leftarrow f^{-1}(f(x_1))\right] \geq \epsilon^2,$$

where the internal random coins used by $\mathcal{A}$ in the two calls are independent.

Proof. For any $y \in Y$, let us define

$$w_y = \mathsf{Prob}\left[f(x) = y \mid x \leftarrow X\right] \quad \text{and}$$
$$\epsilon_y = \mathsf{Prob}\left[P(x, \mathcal{A}(x)) \mid x \leftarrow f^{-1}(y)\right].$$

Then $\sum_{y\in Y} w_y = 1$ and $\sum_{y\in Y} w_y\epsilon_y = \epsilon$. Given the following experiment $x_1 \leftarrow X;\ x_2 \leftarrow f^{-1}(f(x_1))$, then,

$$\begin{aligned}
&\mathsf{Prob}\left[P(x_1,\mathcal{A}(x_1)) \wedge P(x_2,\mathcal{A}(x_2))\right] = \\
&\quad = \sum_{y\in Y} \mathsf{Prob}\left[P(x_1,\mathcal{A}(x_1)) \wedge P(x_2,\mathcal{A}(x_2)) \wedge f(x_1) = y\right] = \\
&\quad = \sum_{y\in Y} \mathsf{Prob}\left[P(x_1,\mathcal{A}(x_1)) \wedge P(x_2,\mathcal{A}(x_2)) \mid f(x_1) = y\right] \mathsf{Prob}\left[f(x_1) = y\right].
\end{aligned}$$

But the condition $f(x_1) = y$ is equivalent to modifying the experiment into $x_1 \leftarrow f^{-1}(y);\ x_2 \leftarrow f^{-1}(y)$. So, in this new probability space, x_1 and x_2 are identically distributed independent random variables, and

$$\begin{aligned}
&\mathsf{Prob}\left[P(x_1,\mathcal{A}(x_1)) \wedge P(x_2,\mathcal{A}(x_2)) \mid f(x_1) = y\right] = \\
&\quad = \left(\mathsf{Prob}\left[P(x_1,\mathcal{A}(x_1)) \mid f(x_1) = y\right]\right)^2 = \epsilon_y^2.
\end{aligned}$$

By using for instance the Cauchy-Schwartz inequality for a suitable weighted inner product (i.e. $\mathbf{a}\cdot\mathbf{b} = \sum_{y\in Y} w_y a_y b_y$), it is straightforward to see that $\sum_{y\in Y} w_y\epsilon_y^2 \geq \epsilon^2$. □

Observe that, although the two calls to $\mathcal{A}$ are not independent, they share part of the input. So, there can be a positive correlation (due to the map f) between their outputs. This is the reason (and not the independence) why the success probability of the two calls can be bounded by the square of the success probability of a single call. Typically, the image of f is a part of the input of $\mathcal{A}$, e.g. when the same RSA modulus is used in both calls to $\mathcal{A}$.

This lemma applies to several security proofs in the literature related to an RSA modulus, where more than one call to an adversary is made.

B Computing the Number of Exceptional Points

In this appendix, we compute the number of points $(\bar{x},\bar{y}) \in \mathbb{Z}_p \times \mathbb{Z}_p^*$ such that $\bar{x} \neq x$ and $\left(\frac{\bar{x}}{\bar{y}}\right)^4 = \left(\frac{x}{y}\right)^4$, where $(x,y) \in D_p$ is the unique point such that $2\#(x,y) = 2\#(\bar{x},\bar{y})$. From observation 1, $(\bar{x},\bar{y}) = (x,y) + (\eta,0)$. Thus

$$\bar{x} = \left(\frac{y}{x-\eta}\right)^2 - x - \eta = \frac{x^3-\eta^3}{(x-\eta)^2} - x - \eta = \frac{x^2+\eta x+\eta^2}{x-\eta} - x - \eta = \eta\frac{x+2\eta}{x-\eta}$$

and

$$\bar{y} = \frac{y}{x-\eta}(\eta-\bar{x}) = \frac{\eta y}{x-\eta}\left(1 - \frac{x+2\eta}{x-\eta}\right) = -\frac{3\eta^2 y}{(x-\eta)^2}.$$

Dividing both equations

$$\frac{\bar{x}}{\bar{y}} = -\frac{(x+2\eta)(x-\eta)}{3\eta y}.$$

On the other hand, $\frac{\bar{x}}{\bar{y}} = \rho\frac{x}{y}$, where ρ is a fourth root of unity. This equation is equivalent to $(x + 2\eta)(x - \eta) = -3\rho\eta x$, that means x is a root of the polynomial equation $(x + 2\eta)^4(x - \eta)^4 = 81\eta^4 x^4$. So, there are at most 8 different values of x, given η. Moreover, there are at most 16 points $(\bar{x}, \bar{y})$ in each curve $E_p(0, b)$ satisfying the conditions at the beginning of this appendix. Finally, the probability that one of these points is guessed at random is at most $16/p$.

A tighter bound for this probability can be obtained if the quadratic equation $(x+2\eta)(x-\eta) = -3\rho\eta x$ is discussed for each value of ρ. Let $t = x/\eta$. The equation can be rewritten as $(t+2)(t-1) = -3\rho t$, and also as $t^2 + (1+3\rho)t - 2 = 0$. The discriminant of the equation is $\Delta = (1 + 3\rho)^2 + 8 = 9\rho^2 + 6\rho + 9$.

Since $p \equiv 1 \bmod 4$, then $\left(\frac{-1}{p}\right) = 1$ and there are 4 different values of ρ: 1, -1 and the two square roots of -1. Moreover, since $p \equiv 5 \bmod 12$, then $\left(\frac{3}{p}\right) = -1$, and $\left(\frac{2}{p}\right) = 1$ if and only if $p \equiv 1 \bmod 8$.

Taking this into account, if $\rho = 1$, then $\Delta = 24$, that is a quadratic residue only if $p \equiv 5 \bmod 8$. If $\rho = -1$, then $\Delta = 12$ that is not a quadratic residue. Finally, if $\rho^2 = -1$, then $\Delta = 6\rho$. But

$$\left(\frac{\rho}{p}\right) = \rho^{\frac{p-1}{2}} = (-1)^{\frac{p-1}{4}} \bmod p$$

that is equal to 1 if and only if $p \equiv 1 \bmod 8$. This implies that 2ρ is always a quadratic residue, so 6ρ never is.

Summing up the above information, the only values of t come up when $p \equiv 5 \bmod 8$ and $\rho = 1$. This two values are $t = -(2 \pm \sqrt{6})$. Now, $x = \eta t$ and $y^2 = x^3 - \eta^3 = \eta^3(t^3 - 1)$. From that, for each value of t, only $\frac{p-1}{2}$ values of η lead to existing values of y. It is easy to see that there are exactly $2(p-1)$ points (x, y), but only $p - 1$ are in D_p.

This last step follows from a symmetry argument. In all equations, (x, y) and $(\bar{x}, \bar{y})$ play a symmetric role, since $(\bar{x}, \bar{y}) = (x, y) + (\eta, 0)$ is equivalent to $(x, y) = (\bar{x}, \bar{y}) + (\eta, 0)$. But $(x, y) \in D_p$ and $(\bar{x}, \bar{y}) \notin D_p$. Thus, only half of the solutions found correspond to values of (x, y), and the other half correspond to values of $(\bar{x}, \bar{y})$.

Advances in Alternative Non-adjacent Form Representations

Gildas Avoine*, Jean Monnerat**, and Thomas Peyrin

EPFL
Lausanne, Switzerland

Abstract. From several decades, *non-adjacent form* (NAF) representations for integers have been extensively studied as an alternative to the usual binary number system where digits are in $\{0, 1\}$. In cryptography, the *non-adjacent digit set* (NADS) $\{-1, 0, 1\}$ is used for optimization of arithmetic operations in elliptic curves. At SAC 2003, Muir and Stinson published new results on alternative digit sets: they proposed infinite families of integers x such that $\{0, 1, x\}$ is a NADS as well as infinite families of integers x such that $\{0, 1, x\}$ is not a NADS, so called a NON-NADS. Muir and Stinson also provided an algorithm that determines whether x leads to a NADS by checking if every integer $n \in [0, \lfloor \frac{-x}{3} \rfloor]$ has a $\{0, 1, x\}$-NAF. In this paper, we extend these results by providing *generators* of NON-NADS infinite families. Furthermore, we reduce the search bound from $\lfloor \frac{-x}{3} \rfloor$ to $\lfloor \frac{-x}{12} \rfloor$. We introduce the notion of *worst* NON-NADS and give the complete characterization of such sets. Beyond the theoretical results, our contribution also aims at exploring some algorithmic aspects. We supply a much more efficient algorithm than those proposed by Muir and Stinson, which takes only 343 seconds to compute all x's from 0 to -10^7 such that $\{0, 1, x\}$ is a NADS.

1 Introduction

It is well known that every positive integer n can be represented as a finite sum of the form $\sum_{i=0}^{n} a_i 2^i$, denoted by $(\ldots a_3 a_2 a_1 a_0)_2$, where the *digits* a_i's are picked in the *digit set* $D = \{0, 1\}$. Using the digit set $\{0, 1\}$ is a common way to represent integers but for some efficiency purposes some alternative digit sets have been proposed during the last decades.

Ternary representations (with radix 3) are mainly due to Lalanne [10] but took off in 1951 when Booth [1] proposed a fast technique to compute the representation of the product of two integers using the $\{-1, 0, 1\}$ radix 2 representation. In 1960, Reitwiesner [17] proved that every integer has a canonical $\{-1, 0, 1\}$

* Supported in part by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

** Supported in part by a grant of the Swiss National Science Foundation, 200021-101453/1.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 260–274, 2004.

radix 2 representation with a minimal number of nonzero digits. This representation called *non-adjacent form* (NAF) is obtained if for any two adjacent digits at least one is zero. Later, in 1989, Jedwab and Mitchell [6] presented an interesting cryptographic application of such representations showing that using the digit set $\{-1, 0, 1\}$ can reduce the number of multiplications in the square-and-multiply algorithm for exponentiation. In elliptic curves, where inversion can be done for (almost) free, exponentiations are much more efficient with such representations. Using this property, Morain and Olivos [14] proposed in 1991 an algorithm speeding up operations over elliptic curves using the $\{-1, 0, 1\}$ digit set. More recently Joye and Tymen [7] proposed a compact encoding of non-adjacent forms applied to elliptic curves, in particular to the Koblitz curves. During the last decade, a certain amount of work has been devoted to non-adjacent form representations such as [2, 3, 4, 5, 8, 9, 12, 13, 18, 19].

In our case, we focus on *ternary* radix 2 representations using the digit set $\{0, 1, x\}$ where x is a negative integer. Determining which sets $\{0, 1, x\}$ provide non-adjacent forms for every positive integer is still an open problem. Such sets are called *non-adjacent digit sets* (NADS). Muir and Stinson [15, 16] gave new results at SAC 2003, proposing some properties that x must verify in order to lead to a NADS and they gave some infinite families of x such that $\{0, 1, x\}$ is or is not a NADS. In the latter case, we say that $\{0, 1, x\}$ is a NON-NADS. They also provided an algorithm that determines whether x is a NADS by checking whether every integer $n \in [0, \lfloor \frac{-x}{3} \rfloor]$ has a $\{0, 1, x\}$-NAF.

We extend in this paper their results by proposing generators that produce infinite families of NON-NADS as much as we wish and we give a way to determine such generators. We reduce also the search bound from $\lfloor \frac{-x}{3} \rfloor$ to $\lfloor \frac{-x}{12} \rfloor$. We introduce the notion of *worst* NON-NADS and give a complete characterization of these numbers. Our contribution aims also at exploring algorithmic aspects related to NADS. So, we propose some improvements of the Muir and Stinson's algorithms [15, 16] that comes from our new theoretical results and we propose a new approach to compute NADS. The first algorithm proposed in [15] took about one day in order to find all x's from -1 to -10^7 such that $\{0, 1, x\}$ is a NADS. While an improved version also due to Muir and Stinson [16] takes about 20 minutes, our own algorithm takes only 343 seconds.

2 Preliminaries and Previous Works

2.1 Definitions and Notation

Every positive integer n can be represented as a finite sum of the form $\sum_{i=0}^{n} a_i 2^i$, denoted by $(\ldots a_3 a_2 a_1 a_0)_2$. Here, the *digits* a_i's are in the *digit set* $D = \{0, 1\}$.

Definition 1. *The Hamming weight of a non-negative integer n, denoted by $w(n)$, is the number of ones in the usual $\{0, 1\}$-radix 2 representation of n.*

Definition 2. *The length of a radix 2 representation* $(\dots a_2 a_1 a_0)_2$ *is the largest integer* ℓ *such that* $a_{\ell-1} \neq 0$ *but* $a_i = 0$ *for all* $i \geq \ell$.

The set of all strings of digits from D is denoted by D^* and contains the empty string ϵ. Every D-radix 2 representation matches a string in D^* and every string in D^* matches a D-radix 2 representation. For $\alpha, \beta \in D^*$, we denote their concatenation by $\alpha \| \beta$. The terminology for representations can be applied to strings. We note $\widehat{\alpha}$ the string formed by deleting the leading zeros from α. For a given digit set D and an integer n, we define the map $R_D(n)$ such that $R_D(n) = \widehat{\alpha}$ where $\alpha \in D^*$ is a D-NAF for n, if one exists, and $R_D(n) = \perp$ otherwise. Here $\perp$ represents the symbol not in D. We are interested in determining which integers have D-NAF's, so we define the set $\mathrm{NAF}(D) := \{n \in \mathbb{Z} : R_D(n) \neq \perp\}$.

Definition 3. *D is a nonadjacent digit set if $\mathbb{Z}^+ \subseteq$ NAF(D).*

2.2 Characterization of NADS

First of all, we give a few theorems, whose proofs can be found in [15], giving necessary conditions for $\{0, 1, x\}$ to be a NADS or a NON-NADS.

Theorem 1. *Let $D = \{0, 1, x\}$. If there exists $n \in$ NAF(D) with $n \equiv 3 \pmod 4$, then $x \equiv 3 \pmod 4$.*

Theorem 2. *The only NADS of the form $\{0, 1, x\}$ with $x > 0$ is $\{0, 1, 3\}$.*

Theorem 3. *If $x \equiv 3 \pmod 4$, then any integer has at most one finite length $\{0, 1, x\}$-NAF form with no leading zeros.*

From Theorems 1 to 3 we see that a $\{0, 1, x\}$-NAF is unique and that x must be negative and congruent to 3 modulo 4 for $\{0, 1, x\}$ to be a NADS (except for $D = \{0, 1, 3\}$). Hence, we only consider NADS such that $x < 0$. The following lemmas lead to an algorithm that determines whether an integer $n \in \mathbb{Z}^+$ has a $\{0, 1, x\}$-NAF.

Lemma 1. *If $n \equiv 0 \pmod 4$ then $n \in$ NAF(D) if and only if $n/4 \in$ NAF(D). Further, if $n \in$ NAF(D) then $R_D(n) = R_D(\frac{n}{4}) \| 00$.*

Lemma 2. *If $n \equiv 1 \pmod 4$ then $n \in$ NAF(D) if and only if $(n-1)/4 \in$ NAF(D). Further, if $n \in$ NAF(D) then $R_D(n) = R_D(\frac{n-1}{4}) \| 01$.*

Lemma 3. *If $n \equiv 2 \pmod 4$ then $n \in$ NAF(D) if and only if $n/2 \in$ NAF(D). Further, if $n \in$ NAF(D) then $R_D(n) = R_D(\frac{n}{2}) \| 0$.*

Lemma 4. *If $n \equiv 3 \pmod 4$ then $n \in$ NAF(D) if and only if $(n-x)/4 \in$ NAF(D). Further, if $n \in$ NAF(D) then $R_D(n) = R_D(\frac{n-x}{4}) \| 0x$.*

We define now the function $f_D : \mathbb{N} \to \mathbb{N}$ as follows: $f_D(n) = \frac{n}{4}$ if $n \equiv 0 \pmod 4$, $f_D(n) = \frac{n-1}{4}$ if $n \equiv 1 \pmod 4$, $f_D(n) = \frac{n}{2}$ if $n \equiv 2 \pmod 4$, $f_D(n) = \frac{n-x}{4}$ if $n \equiv 3 \pmod 4$. For the sake of simplicity, we abuse the notation by denoting $f_0(n) = \frac{n}{4}$, $f_1(n) = \frac{n-1}{4}$, $f_2(n) = \frac{n}{2}$, and $f_3(n) = \frac{n-x}{4}$. Note that $\forall n \in [0, \frac{x}{3}]$, $f_0(n) < n$, $f_1(n) < n$, $f_2(n) < n$, and $f_3(n) > n$. We denote f^i the i-fold composition of the function f. We introduce now the *graph* G_n of an integer n for a given digit set $\{0, 1, x\}$, whose vertices are the iterations of the function f_D on n:

$$n \longrightarrow f_D(n) \longrightarrow f_D^2(n) \longrightarrow f_D^3(n) \longrightarrow \dots$$

where either $\exists k \geq 0$ such that $f_D^k(n) = 0$ or $\exists k_1, k_2,\ 0 \leq k_1 \leq k_2$ such that $f_D^{k_1}(n) = f_D^{k_2}(n)$. In other words, either G_n is a path terminating at 0, or G_n contains a directed cycle of integers in the interval $\{0, 1, 2, \dots, \lfloor \frac{-x}{3} \rfloor\}$ as proved hereafter. The *length of the cycle* is defined as $k_2 - k_1$. Every vertex of G_n is positive. Suppose that $f_D^i(n) < \frac{-x}{3}$, we prove that $f_D^{i+1}(n) < \frac{-x}{3}$. If $f_D^i(n) \equiv 0, 1, 2 \pmod 4$ we have $f_D^{i+1}(n) \leq f_D^i(n) < \frac{-x}{3}$. If $f_D^i(n) \equiv 3 \pmod 4$ then

$$\begin{aligned} f_D^i(n) < \frac{-x}{3} &\implies \frac{f_D^i(n) - x}{4} < \frac{\frac{-x}{3} - x}{4} \\ &\implies f_D^{i+1}(n) < \frac{-x - 3x}{12} = \frac{-x}{3}. \end{aligned}$$

By extension, we define the *graph* $G(x)$ of an integer x as $G(x) := \bigcup_{n=0}^{\lfloor \frac{-x}{3} \rfloor} G_n$. Note that if $\{0, 1, x\}$ is a NADS, then $G(x)$ is a directed tree whose root is 0. We define now the function $g_D : \mathbb{N} \to D^*$ such that: $g_D(n) =$ "00" if $n \equiv 0 \pmod 4$, $g_D(n) =$ "01" if $n \equiv 1 \pmod 4$, $g_D(n) =$ "0" if $n \equiv 2 \pmod 4$, $g_D(n) =$ "0x" if $n \equiv 3 \pmod 4$. From Lemmas 1 to 4 and the definitions of f_D and g_D, Muir and Stinson proposed Lemma 5 that yields Algorithm 1.

Algorithm 1: NAF(n,x)

$\alpha \leftarrow \epsilon$
while $n > \frac{-x}{3}$
do { $\alpha \leftarrow g_D(n) \| \alpha$; $n \leftarrow f_D(n)$ }

$S \leftarrow \varnothing$
while $n \neq 0$
do { **if** $n \in S$ **then return** $\perp$; $S \leftarrow S \cup \{n\}$; $\alpha \leftarrow g_D(n) \| \alpha$; $n \leftarrow f_D(n)$ }

return $\hat{\alpha}$

Algorithm 2: Is-NADS?(x)

$N \leftarrow 3$

while $N \leq \frac{-x}{3}$
do { $n \leftarrow N$; $S \leftarrow \varnothing$; **while** $n \neq 0$ **do** { **if** $n \in S$ **then return** false; $S \leftarrow S \cup \{n\}$; $n \leftarrow f_D\ (n)$ }; $N \leftarrow N + 4$ }

return true

Lemma 5. *For any* $n \in \mathbb{N}$, $n \in NAF(D)$ *if and only if* $f_D(n) \in NAF(D)$. *Further, if* $n \in NAF(D)$ *then* $R_D(n) = R_D(f_D(n))\|g_D(n)$.

The time complexity of the NAF algorithm in the worst case is straightforward: the complexity of the first loop is $O(\log n)$ while the complexity of the second one is $O(|x|)$. The complexity of the algorithm, in the worst case, is therefore $O(\log n + |x|)$. We expose now Theorem 4 that provides algorithm Is-NADS? (See Algorithm 2) determining whether or not a given $x < 0$ leads to a NADS.

Theorem 4. *Suppose* x *is a negative integer and* $x \equiv 3 \pmod 4$. *If every element in the set* $\{n \in \mathbb{Z}^+ : n \leq \lfloor \frac{-x}{3} \rfloor, n \equiv 3 \pmod 4\}$ *has a* $\{0, 1, x\}$*-NAF, then* $\{0, 1, x\}$ *is a NADS.*

The algorithm Is-NADS? requires $O(|x|)$ tests (one test is roughly equivalent to the second loop of the algorithm NAF), therefore the complexity of Is-NADS? is $O(|x|^2)$. Finally, Muir and Stinson [15] give some characterizations of infinite families of NADS and NON-NADS. Among them, we will use the two following theorems.

Theorem 5. *Let* x *be a negative integer with* $x \equiv 3 \pmod 4$. *If* $(2^s - 1) \mid x$ *for any* $s \geq 2$, *then* $\{0, 1, x\}$ *is not a NADS.*

Theorem 6. *Let* x *be a negative integer with* $x \equiv 3 \pmod 4$. *If* $(4 \cdot m_i - 1) < -x < (3 \cdot 2^i)$ *for some* $i \geq 0$, *then* $\{0, 1, x\}$ *is not a NADS, where*

$$m_i := \begin{cases} 2 \cdot \frac{2^i - 1}{3} & \text{for } i \text{ even} \\ \frac{2^{i+1} - 1}{3} & \text{for } i \text{ odd.} \end{cases}$$

3 New Theoretical Results

3.1 Improvement on the Search Domain

By Theorem 4, we know that determining whether $\{0, 1, x\}$ is a NADS can be performed by checking whether every element of the set $\{n \in \mathbb{Z}^+ : n \leq \lfloor \frac{-x}{3} \rfloor, n \equiv 3 \pmod 4\}$ has a $\{0, 1, x\}$-NAF. Here, we prove that the search bound $\lfloor \frac{-x}{3} \rfloor$ can be improved when 3 or/and 7 do not divide x. So, Theorem 7 reduces the bound to $\lfloor \frac{-x}{6} \rfloor$ and Theorem 8 goes further reducing the search domain to $]0, \lfloor \frac{-x}{12} \rfloor] \cup [\lfloor \frac{-x}{7} \rfloor, \lfloor \frac{-x}{6} \rfloor]$.

Theorem 7. *Let* x *be a negative integer such that* $3 \nmid x$ *and* $x \equiv 3 \pmod 4$. *If every element in the set* $\{n \in \mathbb{Z}^+ : n \leq \lfloor \frac{-x}{6} \rfloor, n \equiv 3 \pmod 4\}$ *has a* $\{0, 1, x\}$*-NAF, then* $\{0, 1, x\}$ *is a NADS.*

Proof. Let $n \equiv 3 \pmod 4$ be a positive integer such that $n \leq \lfloor \frac{-x}{3} \rfloor$. Since $3 \nmid x$, $n < \frac{-x}{3}$. We have to show that G_n must contain at least one vertex which is less than $\frac{-x}{6}$. In other words, this corresponds to show the existence of $j \in \mathbb{N}$ satisfying $f_D^j(n) < \frac{-x}{6}$. From definition of f_D, we also remark that if $f_D^i(n) < \frac{-x}{3}$ is congruent to $0, 1, 2$ modulo 4, then $f_D^{i+1}(n) < \frac{-x}{6}$. So, it remains to show that $f_D^i(n)$ cannot be congruent to 3 modulo 4 for all $i \in \mathbb{N}$. From Section 2.2, we know that $f_D^i(n) < \frac{-x}{3} \Rightarrow f_D^{i+1}(n) < \frac{-x}{3}$ for $i \in \mathbb{N}$ and that f_D is strictly increasing on the upper bounded set $\{n \in \mathbb{Z}^+ : n \leq \frac{-x}{3}, n \equiv 3 \pmod 4\}$. Hence, $f_D^i(n)$ cannot be congruent to 3 modulo 4 for all $i \in \mathbb{N}$. □

We give here a further improvement on the search domain.

Theorem 8. *Let x be a negative integer such that $3 \nmid x$, $7 \nmid x$ and $x \equiv 3 \pmod 4$. If every element in the set $\{n \in \mathbb{Z}^+ : n \leq \lfloor \frac{-x}{12} \rfloor, n \equiv 3 \pmod 4\} \bigcup \{n \in \mathbb{Z}^+ : \lfloor \frac{-x}{7} \rfloor \leq n \leq \lfloor \frac{-x}{6} \rfloor, n \equiv 3 \pmod 4\}$ has a $\{0, 1, x\}$-NAF, then $\{0, 1, x\}$ is a NADS.*

Proof. Let n be a positive integer such that $n \equiv 3 \pmod 4$ and $\lfloor \frac{-x}{12} \rfloor \leq n \leq \lfloor \frac{-x}{7} \rfloor$. We will show that G_n contains at least a vertex that lies in the interval $[\frac{-x}{7}, \frac{-x}{6}]$ or that is less than $\frac{-x}{12}$. First, we notice that if an element of G_n is congruent to 0 or 1 modulo 4, then this one will be sent to an integer less than $\frac{-x}{12}$ since this element cannot be greater than or equal $\frac{-x}{3}$. So, it remains to consider the n's for which G_n contains only vertices congruent to 2 or 3 modulo 4. Given that $f_2 \circ f_2 = f_0$, such a n is transformed by iterations of the form

$$f_2 \circ f_3^{i_k} \circ f_2 \circ f_3^{i_{k-1}} \circ \cdots f_2 \circ f_3^{i_1}, \tag{1}$$

for some integer $k \geq 1$ and where $i_1, \ldots, i_k$ are positive integers. We set $F := f_2 \circ f_3$. We see that $F(n) = \frac{n-x}{8}$ and that $F(n) > n \Leftrightarrow n < \frac{-x}{7}$. From the properties of the function f_3, we can conclude that $f_2 \circ f_3^\ell(n) \geq F(n)$ for $\ell \in \mathbb{N}$ and $n \leq \frac{-x}{3}$. Hence, the value of some iterations of the form (1) applied to n is greater or equal to $F^k(n)$. We finally deduce that there exists a positive integer k such that the resulting integer n' of the iteration (1) applied on n is greater than $\frac{-x}{7}$, since the intermediate value increases after each iteration of a function of the form $f_2 \circ f_3^\ell$ and since $7 \nmid x$. Moreover, n' is less than $\frac{-x}{6}$ since every values of G_n are less than $\frac{-x}{3}$ and that the last operation of (1) is a division by 2.

□

Conjecture 1. Let x be a negative integer such that $3 \nmid x$, $7 \nmid x$ and $x \equiv 3 \pmod 4$. If every element in the set $\{n \in \mathbb{Z}^+ : n \leq \lfloor \frac{-x}{12} \rfloor, n \equiv 3 \pmod 4\}$ has a $\{0, 1, x\}$-NAF, then $\{0, 1, x\}$ is a NADS.

The new results presented in this section are particularly important because they allow to reduce significantly the running time of Algorithm 2, as we will see in Section 4.

3.2 Generators of Infinite Families of NON-NADS

In this section, we present a way to generate as many NON-NADS families as we want. From a theoretical point of view, this method allows to find all NON-NADS. In practice, it will be used as a trade-off in our algorithm Find-NADS (See Section 4) that computes every x such that $\{0, 1, x\}$ is a NADS.

The idea of our method comes from the fact that $n \in \mathrm{NAF}(D)$ if and only if G_n does not contain any directed cycle. So, the existence of an integer n such that G_n contains a directed cycle implies that D is not a NADS. Instead of looking for a criteria on x for which there exists such a n, we consider a cycle of a given form and deduce the values x for which n lies in this cycle. More precisely, we choose the length t of the cycle as well as the sequence of the t different functions f_i for $i \in \{0, 1, 2, 3\}$ that are applied successively on n. Once the form of the cycle is chosen, we set for a positive integer $n \equiv 3 \pmod 4$ the equation

$$f_D^t(n) = f_{i_t} \circ f_{i_{t-1}} \circ \ldots f_{i_1}(n) = n, \tag{2}$$

where $i_k \in \{0, 1, 2, 3\}$ for $k = 1, 2 \ldots, t$. We denote such a cycle of length t as $i_1|i_2|\ldots|i_t$. From (2), we obtain a relation of the form $c_1 n = c_2 x$ for two given $c_1, c_2 \in \mathbb{Z}$. It remains to substitute $n = 4k - 1$ in this equation and solve it with the conditions that $k \in \mathbb{N}$ and x is negative with $x \equiv 3 \pmod 4$. Note that $i_1 = 3$.

2-Cycles. To illustrate our method, we show how we can concretely find every cycle of length 2. Such a cycle is called a 2-cycle. First, we observe that we have 3 possible 2-cycles, namely $3|0$, $3|1$ and $3|2$. They correspond to the equations $\frac{n-x}{16} = n$, $\frac{n-x-4}{16} = n$ and $\frac{n-x}{8} = n$. The first equation provides $x = -15n$ and since $n = 4k - 1$, we then have $x = -60k + 15$ for $k \in \mathbb{N}$. By setting $k = 7$, we see that $n = 27$, $x = -405$ and $f_3(27) = 108 = 4 \cdot 27$. The second equation provides $x = -15n - 4 = -60k + 11$. Similarly, we obtain $x = -28k + 7$ from the third equation.

Theorem 9. *If $x = -60k + 15$, $x = -60k + 11$ or $x = -28k + 7$ with $k \in \mathbb{N}$, then $\{0, 1, x\}$ is a NON-NADS.*

Some Cycles of Arbitrary Length. Here, we apply our method to find an infinite number of NON-NADS families. As an illustration, we look for the x's whose graph $G(x)$ contain a cycle of the form $3|3|3|\ldots|3|0$. Let $t \geq 2$ be the length of this cycle, we have to solve $f_0 \circ f_3^{t-1}(n) = n$. Let us first compute $f_3^{t-1}(n)$. We have

$$f_3^{t-1}(n) = \frac{n - x\sum_{i=1}^{t-1} 4^{i-1}}{4^{t-1}} = \frac{n - \frac{x(4^{t-1}-1)}{3}}{4^{t-1}}$$

and hence we get the equation

$$f_0 \circ f_3^{t-1}(n) = \frac{n - \frac{x(2^{2t-2}-1)}{3}}{2^{2t-1}} = n.$$

This holds if and only if $-x(2^{2t-2}-1)/3 = n(2^{2t-1}-1)$. From this, x has to be a multiple of $(2^{2t-1}-1)$ since $\gcd((2^{2t-2}-1)/3, 2^{2t-1}-1) = 1$. Moreover, $x \equiv 3 \pmod 4$ implies that x is of the form $x = -(4k-1)(2^{2t-1}-1)$ for $k \in \mathbb{N}$. We can also see that $n = (4k-1)(4^{t-1}-1)/3$ is congruent to 3 modulo 4 and that it is positive.

Theorem 10. *Let $t \geq 2$ and $k > 0$ be two integers and $x = -(4k-1)(2^{2t-1}-1)$. Then $\{0, 1, x\}$ is a NON-NADS.*

Note that for $t = 2$, this generates the x-family corresponding to that of Theorem 9, namely $-28k + 7$. Obviously, if we consider cycles of another form (instead of $3|3|3|\ldots|3|0$) we obtain some other generators.

3.3 Worst NON-NADS

We introduce in this section the notion of *worst NON-NADS* and give a complete characterization of it.

Definition 4. *Let x be a negative integer such that $x \equiv 3 \pmod 4$. $\{0, 1, x\}$ is a worst NON-NADS if for all $n \leq -\frac{x}{3}$ with $n \equiv 3 \pmod 4$, $n \notin NAF(\{0, 1, x\})$.*

Theorem 11. *Let x be a negative integer such that $x \equiv 3 \pmod 4$. $\{0, 1, x\}$ is a worst NON-NADS if and only if there exists $i \geq 2$ such that $(4m_i - 1) < -x < (3 \cdot 2^i)$, where*

$$m_i := \begin{cases} 2 \cdot \frac{2^i-1}{3} & \text{for } i \text{ even} \\ \frac{2^{i+1}-1}{3} & \text{for } i \text{ odd} \end{cases}$$

Proof. We first prove that if a given x is in an interval of the form $]-3\cdot 2^i, 1-4m_i[$ then $\{0, 1, x\}$ is a worst NON-NADS. Next we prove that if $\{0, 1, x\}$ is a worst NON-NADS then it is in such an interval. Such an interval is called a *gap*. The first part of the proof directly comes from the proof of Theorem 21 of [16].

We prove now the converse statement. In other words, we show that for each x which is not in a gap, there exists a n such that $n \in \mathrm{NAF}(\{0, 1, x\})$. We introduce the notion of *pivot*: x_p is a pivot if there exists a $i \geq 2$ s.t. $x_p = 3 - 2^{i+2}$.

We prove that, for all $i \geq 2$, there is no worst NON-NADS in $[3 - 2^{i+2}, -3 \cdot 2^i - 1]$. Let x_p be the pivot $3 - 2^{i+2}$; we have

$$f_{\{0,1,x_p\}}(3) = \frac{3 - (3 - 2^{i+2})}{4} = 2^i,$$

implying that $3 \in \mathrm{NAF}(\{0, 1, x_p\})$. We have furthermore $f_{\{0,1,x_p+4k\}}(3 + 4k) = f_{\{0,1,x_p\}}(3)$ for all integers $k \geq 1$. Therefore, $x_p + 4k$ is a worst NON-NADS if

$$3 + 4k < \left\lfloor \frac{-x_p - 4k}{3} \right\rfloor < \frac{-x_p - 4k}{3} - 1.$$

Hence, we can compute that this inequality holds if and only if $k < 2^{i-2} - \frac{15}{16}$. This implies that for every $x \in [3 - 2^{i+2}, -3 \cdot 2^i - 1]$, we have found an $n < \lfloor -\frac{x}{3} \rfloor$ that is a $\{0, 1, x\}$-NAF.

It remains to prove that the interval $I_i := [1 - 4m_{i+1}, 3 - 2^{i+2}]$ does not contain any integer $x \equiv 3 \pmod 4$ that is a worst NON-NADS. To this end, we first show that 3 is a $\{0, 1, x\}$-NAF for the smallest $x \in I_i$, i.e., for $x = 3 - 4m_{i+1}$. Indeed, for an odd i, it suffices to see that $3 = (101010 \ldots 100x)_2$, where the sequence 01 is repeated $(i + 1)/2$ times. This is shown by the following computation

$$(1010 \ldots 100x)_2 = 3 - 4m_{i+1} + \sum_{j=1}^{\frac{i+1}{2}} 2^{2j+1} = \frac{17 - 2^{i+4}}{3} + 8 \cdot \frac{2^{i+1} - 1}{3} = 3.$$

We deduce that for $x_k = 4k + x$, where $k \geq 1$, we also have $(1010 \ldots 100x_k)_2 = 3 + 4k$. Moreover, we can also show that $3 + 4k < \lfloor -\frac{x_k}{3} \rfloor$ for all $k \leq \frac{2^i - 2}{3}$ and that these x_k's correspond to all elements of the interval I_i that are congruent to 3 modulo 4. This proves that the intervals I_i's for the odd integers i's do not contain any worst NON-NADS. The case where i is even can be proved in the same way by showing that $3 = (0101 \ldots 010x)_2$, where the sequence 01 occurred $\frac{i}{2} + 1$ times. This concludes our proof. □

4 Algorithmic Considerations

We use in this section the theoretical results presented in Section 3 combined with some algorithmic methods in order to reduce the running time of the NADS search. First of all, we recall the basic algorithm (Algorithm 3) proposed by Muir and Stinson [15] and then we bring some improvements that greatly improve the performances. So, Section 4.2 takes benefit of the theoretical results of Section 3.1. Results presented in Section 3.2 are on their hand used in Section 4.3. We then give the performances of our best algorithm in Section 4.4 and show that when $x_{\max} = -10^7$, the running time of our algorithm is only 343 seconds.

Algorithm 3: Find-NADS $(x_{\max})$

NADS $\leftarrow \varnothing$

for $i = -1$ **to** $i = x_{\max}$
do { **if** (Is-NADS? i) **then** NADS $\leftarrow$ NADS $\cup \{i\}$; $i \leftarrow i - 4$

return NADS

4.1 Basic Algorithm

The algorithm Find-NADS (Algorithm 3) is the algorithm proposed by Muir and Stinson. It finds NADS from -1 to $x_{\max}$, iterating on this interval the algorithm Is-NADS? presented in Section 2.2 which aims at determining whether or not a given negative x leads to a NADS. Its performances are given in Section 4.4.

4.2 Intra-X and Inter-X Techniques

The *intra-X technique* consists of using memoization method during the execution of Is-NADS?. Memoization is an optimization technique whose basic idea is to remember function calls. A table is maintained that maps lists of argument values to previously computed return values for those arguments. When a function is called, its list of arguments is looked up in the table. If an entry is found, then the previously computed value is returned directly. Otherwise, the value is computed and then stored in the table for future use. Such a technique is well-suited for Algorithm 2 since function f_D is called many times with the same argument.

The *Inter-X technique* is an extension of the Intra-X technique using memoization during the execution of Find-NADS. Note however that the return value of f_D depends on both n and x. However, the result of $f_D(n)$ is independent of x when $n \not\equiv 3 \pmod 4$. The intuitive idea consists roughly in establishing shortcuts between n and successive iterations $f_D^k(n)$ until reaching a value congruent to 3 modulo 4. We give hereafter a formal approach, by introducing equivalence classes representing such shortcuts. Let b be the function from $\mathbb{N}$ to $\mathbb{N}$ defined by $b(n) = f_D^k(n)$ where $k \geq 0$, $f_D^k(n) \equiv 3 \pmod 4$ or $f_D^k(n) = 0$, and $\forall k'\ 0 \leq k' < k$, $f_D^{k'}(n) \not\equiv 3 \pmod 4$ and $f_D^{k'}(n) \neq 0$. We define the equivalence relation $\mathcal{R}$ such that $n\mathcal{R}n' \Leftrightarrow b(n) = b(n')$. The equivalence class of n induced by $\mathcal{R}$, denoted by $\dot{n}$ is therefore the set $\{n' \in \mathbb{N} \mid n\mathcal{R}n'\}$. The smallest element of $\dot{n}$ is called the *representative* of the class. Any element of $\dot{n}$ has a $\{0, 1, x\}$-NAF if and only if the representative of $\dot{n}$ has a $\{0, 1, x\}$-NAF. As illustration, we give on Fig. 1 some elements of the class whose representative is 7. The equivalence classes of 0 and all

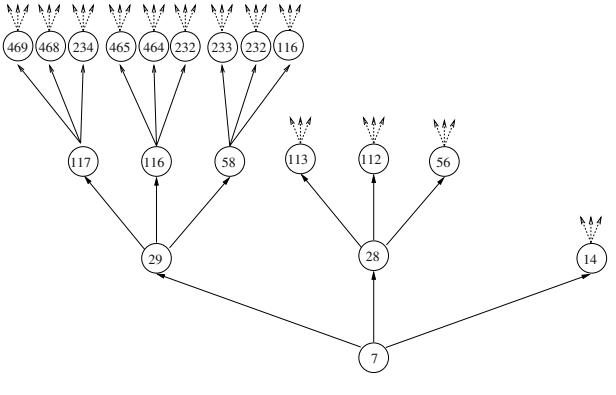

Fig. 1. Class of equivalence whose representative is 7

n such that $n \equiv 3 \pmod 4$ are therefore pre-computed and stored in a table. This pre-computation is lightweight: for every $n \equiv 3 \pmod 4$ in the interval $[0, \frac{-x}{3}]$, the inverse of the procedure f_D is recursively applied until $n > \frac{-x}{3}$. The algorithm Classes given in the appendix is the pseudo-code of this pre-computation stage. When this pre-computation is achieved, each cell of the table, indexed with n, contains the representative of $\dot{n}$. So, algorithm Is-NADS? uses the function $f_D(n)$ only when $n \equiv 3 \pmod 4$ and looks up the value in the table otherwise. Note that the table induced only a low complexity space, that is $O(|x| \log |x|)$.

4.3 Algorithm Based on Elimination of NON-NADS

We present in this section an algorithm, based on a new approach, that consists of finding all the x leading to a NADS by process of elimination of all NON-NADS. This algorithm, Elim-NON-NADS, relies on the theoretical results presented in Section 3.2. The rough idea of this algorithm is to eliminate all NON-NADS lower than a given bound $x_{\max}$ having a cycle of length t, where t varies from 1 to $\lfloor \frac{-x_{\max}}{3} \rfloor$. Indeed, x is a NON-NADS if and only if $\exists n \in \mathbb{N}, \exists t \geq 2$ such that $f_D^t(n) = n$. For instance, the cycle 3|0 yields the equation $\frac{n-x}{16} = 16$ that is $-x = 3n$. By iterating t, we can obtain all the possible values of x that reach a cycle by using a depth-first search in the the exploration tree of the different ways to construct a cycle. Using results of Section 3.2, we obtain:

$$-x = \frac{n \cdot (k_1 - 1) + k_3}{k_2} \text{ with } k_1 \geq 4,\ k_2 \geq 1,\ k_3 \geq 0,\ n \geq 3.$$

We move in the tree using the following formulas: $k_1 = 4k_1$, $k_2 = k_2$, $k_3 = k_3$, if $n \equiv 0 \pmod 4$; $k_1 = 4k_1$, $k_2 = k_2$, $k_3 = k_3 + k_1$ if $n \equiv 1 \pmod 4$; $k_1 = 2k_1$, $k_2 = k_2$, $k_3 = k_3$ if $n \equiv 2 \pmod 4$; and $k_1 = 4k_1$, $k_2 = k_2 + k_1$, $k_3 = k_3$ if $n \equiv 3 \pmod 4$. In practice, this algorithm does not allow to find all the NON-NADS when x is large due to the exponential time complexity of the tree exploration. However, it can be used to reduce the time complexity of Find-NADS by finding all NON-NADS that have cycles of length lower or equal to $t_{\max}$ such that $t_{\max}$ is small enough. Indeed, determining all NON-NADS having small cycles is much more faster with Elim-NON-NADS than with the basic Find-NADS. Consequently, finding all NON-NADS can be improved using a trade-off between Elim-NON-NADS and the basic Find-NADS. $t_{\max}$ is the *parameter* of the trade-off. As described in the appendix, Find-NADS uses Elim-NON-NADS as a sieve in a first stage in order to rough out the search process.

4.4 Experimental Results and Memory Complexity

We give in this section some experimental results in order to compare the performances of the presented algorithms. The tests were done on a standard workstation. We experimented the following algorithms whose results are given in Table 1 and represented in Fig. 2.

1. Curve A: the basic algorithm [15]; we ran the C source code that the authors gracefully provided to us.

2. Curve B: the improved basic algorithm [16]; we implemented ourself the algorithm Is-NADS? provided in [16]. We implemented then the algorithm Find-NADS using a sieve to eliminate NON-NADS characterized by Theorem 5 initially proposed in [11] and Theorem 6.
3. Curve C: our algorithm, takes benefit of our new theoretical results presented in Sections 3.1 and 3.2, and practical results described in Sections 4.2 and 4.3. It is actually a trade-off of parameter $t_{\max} = 10$ between the improved version of Find-NADS and Elim-NON-NADS. The pseudo-code of this algorithm is given in the appendix.

Note that the three implementations have been compared in a fair way (as much as possible). They have been implemented in C, compiled with the same optimization options, and executed on the same AMD Athlon^{TM} XP2500+ processor. We did not try to minimize the running time of the algorithms by using some special low level functions of the language. We would like to emphasize that the memory complexity in the worse case, that is when all the cycle lengths equal

Table 1. Running time of the experimented algorithms

$x_{\max}$	Running time (seconds)		
	Basic algorithm (curve A)	Improved basic algorithm (curve B)	Our algorithm (curve C)
-10^5	7	1	<1
-10^6	655	15	3
$-3 \cdot 10^6$	1550	137	35
$-6 \cdot 10^6$	6132	435	127
$-10 \cdot 10^6$	68532	1154	343
$-13 \cdot 10^6$	–	1460	438
$-17 \cdot 10^6$	–	2454	724

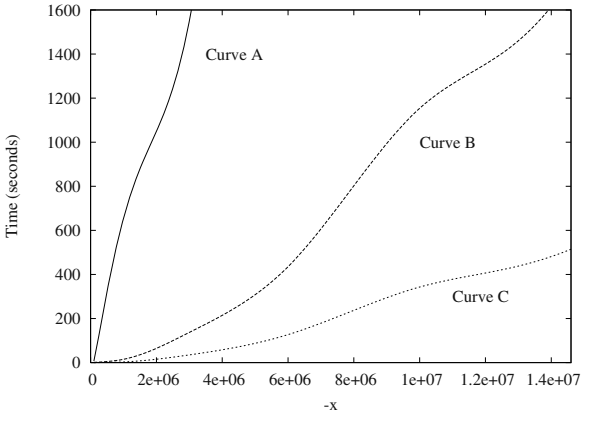

Fig. 2. Running time of the experimented algorithms

$\lfloor \frac{-x}{3} \rfloor$, is the same whatever the algorithm is. Indeed, the memory complexity is in the worse case $O(|x| \log |x|)$. Our algorithm requires however slightly more memory on average due to the precomputation steps. It fits nevertheless into a quite small memory since it requires only a few tens of megabytes of RAM.

5 Conclusion

We extended in this paper previous works mainly done by Muir and Stinson [15, 16]. Our main contribution consists of a method providing *generators* of NON-NADS infinite families and a reduction of the search domain to the interval $[0, \lfloor \frac{-x}{12} \rfloor] \cup [\lfloor \frac{-x}{7} \rfloor, \lfloor \frac{-x}{6} \rfloor]$ when x is not divided by 3 and 7. We claimed that we can still reduce it to $[0, \lfloor \frac{-x}{12} \rfloor]$. We also introduced the notion of worst NON-NADS and characterized them. From these new theoretical results, we suggested some algorithmic improvements that reduce significantly the running time of the algorithm Find-NADS. Our algorithm takes only 343 seconds when $x_{\max} = -10^7$, while the best known algorithm [16] took about 20 minutes.

References

1. A. Booth. A signed binary multiplication technique. *The Quarterly Journal Mechanics and Applied Mathematics*, 4:236–240, 1951.
2. W. Bosma. Signed bits and fast exponentiation. *Journal de théorie des nombres de Bordeaux*, 13(1):27–41, 2001.
3. E. De Win, S. Mister, B. Preneel, and M. Wiener. On the performance of signature schemes based on elliptic curves. In J. Buhler, editor, *Algorithmic Number Theory, ANTS-III*, LNCS **1423**, pp. 252–266, USA, 1998. Springer.
4. K. Fong, D. Hankerson, J. López, and A. Menezes. Field inversion and point halving revisited. *IEEE Transactions on Computers*, 53(8):1047–1059, August 2004.
5. D. Gordon. A survey of fast exponentiation methods. *Journal of Algorithms*, 27(1):129–146, 1998.
6. J. Jedwab and C. Mitchell. Minimum weight modified signed-digit representations and fast exponentiation. *Electronics Letters*, 25(17):1171–1172, 1989.
7. M. Joye and C. Tymen. Compact encoding of non-adjacent forms with applications to elliptic curve cryptography. In K. Kim, editor, *PKC 2001*, LNCS **1992**, pp. 353–364, Korea, 2001. Springer.
8. M. Joye and S. Yen. Optimal left-to-right binary signed-digit recoding. *IEEE Transactions on Computers*, 49(7):740–748, 2000.
9. K. Koyama and Y. Tsuruoka. Speeding up elliptic cryptosystems by using a signed binary window method. In E. Brickell, editor, *CRYPTO'92*, LNCS **740**, pp. 345–357, USA, 1992. IACR, Springer.
10. L. Lalanne. Note sur quelques propositions d'arithmologie élémentaire. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 11:903–905, 1840.
11. D. Matula. Basic digit sets for radix representation. *Journal of the Association for Computing Machinery*, 29(4):1131–1143, 1982.
12. K. Okeya and T. Takagi. The Width-w NAF Method Provides Small Memory and Fast Elliptic Scalar Multiplications Secure against Side Channel Attacks. *CT-RSA*, LNCS **2612**, pp. 328–343, USA, 2003. Springer.

13. K. Okeya and T. Takagi. A More Flexible Countermeasure against Side Channel Attacks Using Window Method. *CHES 2003*, LNCS **2779**, pp. 397–410, Germany, 2003. IACR, Springer.
14. F. Morain and J. Olivos. Speeding up the computations on an elliptic curve using addition-subtraction chains. *RAIRO – Theoretical Informatics and Applications*, 24(6):531–543, 1990.
15. J. Muir and D. Stinson. Alternative digit sets for nonadjacent representations. In M. Matsui and R. Zuccherato, editors, *SAC 2003*, LNCS **3006**, pp. 306–319, Canada, 2003. IACR, Springer.
16. J. Muir and D. Stinson. Alternative digit sets for nonadjacent representations. Technical Report CORR 2004-09, Centre for Applied Cryptographic Research, University of Waterloo, Canada, 2004, `http://www.cacr.math.uwaterloo.ca`.
17. G. Reitwiesner. Binary arithmetic. *Advances in Computers*, 1:231–308, 1960.
18. J. Solinas. An improved algorithm for arithmetic on a family of elliptic curves. In B. Kaliski, editor, *Crypto'97*, LNCS **1294**, pp. 357–371, USA, 1997. IACR, Springer.
19. J. Solinas. Efficient arithmetic on Koblitz curves. *Designs, Codes and Cryptography*, 19(2–3):195–249, 2000.

Appendix: The Final Algorithms

Algorithm: Find-NADS($x_{\max}$, $t_{\max}$)

NADS $\leftarrow \varnothing$
Classes($x_{\max}$)
Elim-NON-NADS($x_{\max}, t_{\max}$)

for $i = -1$ **to** $i = x_{\max}$
do { **if** (Is-NADS? (i))
  then NADS $\leftarrow$ NADS $\cup \{i\}$
$i \leftarrow i - 4$

return NADS

Algorithm: Is-NADS?(x)

$N \leftarrow 3$
$T \leftarrow \varnothing$
while $N < \frac{-x}{12}$
do {
  $n \leftarrow N$
  $S \leftarrow \varnothing$
  while $n \neq 0$
  do {
    if $n \in S$
      then return false
    if $n \in T$
      then break
    $S \leftarrow S \cup \{n\}$
    $T \leftarrow T \cup \{n\}$
    if $n \equiv 3 \pmod 4$
      then $n \leftarrow \frac{n-x}{4}$
    else
      then $n \leftarrow V[n]$
  $N \leftarrow N + 4$

return true

Fig. 3. Final algorithms

Algorithm: Classes($x_{\max}$)

for $i = 3$ **to** $i = \frac{-x_{\max}}{3}$
do { Classes-rec($x_{\max}, i, i$); $i \leftarrow i + 4$

Algorithm: Classes-rec($x_{\max}, n_{\text{seed}}, n_{\text{cur}}$)

$V[n_{\text{cur}}] \leftarrow n_{\text{seed}}$

if $n_{\text{cur}} < \frac{-x_{\max}}{3}$
{ Classes-rec($x_{\max}, n_{\text{seed}}, 2 \cdot n_{\text{cur}}$)
Classes-rec($x_{\max}, n_{\text{seed}}, 4 \cdot n_{\text{cur}}$)
Classes-rec($x_{\max}, n_{\text{seed}}, 4 \cdot n_{\text{cur}} + 1$)

Fig. 4. Precomputation: the classes of equivalence

Algorithm: Elim-NON-NADS($x_{\max}, t_{\max}$)

Elim-NON-NADS-rec($x_{\max}, 4, 1, 0$, "0", $1, t_{\max}$)

Algorithm: Elim-NON-NADS-rec($x_{\max}, k_1, k_2, k_3, e_{\text{prev}}, t_{\text{cur}}, t_{\max}$)

if $e_{\text{prev}} \neq$ "0x"
{ **for** $i = 3$ **to** $i = \frac{-x_{\max}}{6}$
do { $x_{\text{cur}} = \frac{-i \cdot (k_1 - 1) + k_3}{k_2}$
if (x_{cur} is integer) **and** ($x_{\text{cur}} \equiv 3 \pmod 4$)
then $B[x_{\text{cur}}] \leftarrow$ true
$i \leftarrow i + 4$

if $t_{\text{cur}} \leq t_{\max}$
{ **if** $e_{\text{prev}} \neq$ "0"
{ Elim-NON-NADS-rec($x_{\max}, 4 \cdot k_1, k_2, k_3$, "00", $t_{\text{cur}} + 1, t_{\max}$)
Elim-NON-NADS-rec($x_{\max}, 4 \cdot k_1, k_2, k_3 + k_1$, "01", $t_{\text{cur}} + 1, t_{\max}$)
Elim-NON-NADS-rec($x_{\max}, 2 \cdot k_1, k_2, k_3$, "0", $t_{\text{cur}} + 1, t_{\max}$)
Elim-NON-NADS-rec($x_{\max}, 4 \cdot k_1, k_2 + k_1, k_3$, "0$x$", $t_{\text{cur}} + 1, t_{\max}$)
else
{ Elim-NON-NADS-rec($x_{\max}, 4 \cdot k_1, k_2, k_3 + k_1$, "01", $t_{\text{cur}} + 1, t_{\max}$)
Elim-NON-NADS-rec($x_{\max}, 4 \cdot k_1, k_2 + k_1, k_3$, "0$x$", $t_{\text{cur}} + 1, t_{\max}$)

Fig. 5. Precomputation: the sieve

Attacks on Public Key Cryptosystems Based on Free Partially Commutative Monoids and Groups

Françoise Levy-dit-Vehel and Ludovic Perret

ENSTA, 32 Boulevard Victor, 75739 Paris cedex 15
{levy, lperret}@ensta.fr

Abstract. At INDOCRYPT 2003, Abisha, Thomas and Subramanian have proposed a public key encryption scheme and a zero-knowledge authentication protocol based on the word problem on monoids, as well as a group variant of these systems. We here present a total break attack on each of the two encryption schemes. The complexity bounds of our algorithms show that these schemes are insecure for practical parameter sizes. In the monoid setting, we go one step further by proposing an algorithm that breaks the NP-hard problem underlying both the encryption scheme and the zero-knowledge protocol, as well as an upper bound on its complexity.

Keywords: Public Key Cryptanalysis, Word Problem, Thue Systems, Finitely Presented Groups, Free Partially Commutative Monoids, Homomorphic Mappings.

1 Introduction

Following the need for alternatives to number-theoretic based public key cryptosystems, Abisha, Thomas and Subramanian have designed in [1] encryption schemes and zero-knowledge authentication protocols based on the word problem in monoids and groups. In a very general setting, the word problem is that of deciding whether two words x and y over a finite alphabet Σ are equivalent with respect to some set of relations $R \subset \Sigma^* \times \Sigma^*$. This problem has been proven undecidable for general instances (Σ, R) [14]. This problem, and more generally the algebra of string rewriting systems admit a broad spectrum of applications in computer science such as formula-manipulation systems, parallel computation, or modeling concurrency control problems in data base systems [3, 4, 2]. In the cryptographic context, several attempts have been made in the past to design secure public key encryption schemes from this or related problems (e.g. [15, 13, 11], and more recently [12, 10]), but most succumbed to cryptanalysis [6, 9].

In this paper, we present an attack on each of the two public key encryption schemes proposed in [1]: the one based on the word problem on monoids, that we shall call FPCM system in the sequel, and its group-based counterpart, that

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 275–289, 2004.

we name FPCG[1]. Both systems are constructed as follows: the public key is a finitely presented monoid (resp. group) where the word problem is hard, and the secret key is a free partially commutative monoid (resp. group) where the word problem is known to be easy, together with a morphism mapping from one structure to the other.

To break FPCM and FPCG systems, our idea is to observe that it suffices to focus on the finding of a suitable morphism; the free partially commutative structure then follows. We then make use of the fact that the morphisms considered to design these systems are of a special form.

In the monoid context, we go one step further by proposing a method to break the hard problem underlying FPCM, that we call TMMI problem (Thue Monoid Morphism Interpretation problem). Our method relies on a very nice property of congruence relations in free partially commutative monoids due to R. Cori and D. Perrin [5]. Thus, our attack also breaks the zero-knowledge authentication protocol of [1].

It has to be mentioned that, soon after the publication of [1], a chosen ciphertext attack [8] has been proposed, that breaks both of the encryption schemes[2] by means of $O(|\Delta|^2)$ queries to a decryption oracle (Δ being the public alphabet). Although this attack is very efficient, our approach has a wider scope, in the sense that it focuses on the underlying hard problem, as well as its particular instances. The attacks we propose, still exponential in $|\Delta|$, are fast enough to compromise the use of practical sizes of Δ.

The paper is organized as follows: the next section introduces the necessary material to present the monoid-based FPCM system. Section 3 will describe this system and state the associated underlying hard problem TMMI, from which the FPCM instances constitute a particular class. We follow by a description of the algorithm for cryptanalysing FPCM (section 4). In section 5, we present our method to break the general instances of TMMI (section 5), as well as a synthetic algorithmic translation of it (algorithm 2). Next (section 6), we turn to the group variant of the FPCM system, namely the FPCG system. Adapting the ideas presented before to this setting, we propose another algorithm that breaks FPCG. For all the algorithms presented, we give upper bounds on their complexity. Some concluding remarks end the paper.

2 Terminology and Notations

Let Δ be a finite alphabet, and Δ^* be the free monoid over Δ, with λ representing the empty word. When dealing with strings, the appropriate notion of rewriting systems is that of a *Thue system*, namely a subset T of $\Delta^* \times \Delta^*$. We here only consider *finite* Thue systems, i.e. T is a finite set of pairs of strings. Each pair $(\ell, r) \in T$ is called a *rule*. The *single step reduction relation* on Δ^* induced by T

[1] FPCM stands for free partially commutative monoid, and FPCG stands for free partially commutative group.

[2] But not the zero-knowledge protocols of course.

is defined as follows: for any $s, t \in \Delta^*$, $s \leftrightarrow_T t \iff$ there exist $x, y \in \Delta^*$ and $(\ell, r) \in T$ such that ($s = x\ell y$ and $t = xry$) or ($s = xry$ and $t = x\ell y$). The *Thue congruence*, denoted by $\overset{*}{\leftrightarrow}_T$, generated by T is the reflexive transitive closure of $\leftrightarrow_T$. Two words $u, v \in \Delta^*$ are *congruent with respect to* T iff $u \overset{*}{\leftrightarrow}_T v$.

Note that the set T is a presentation of the so-called *finitely presented monoid* $\Delta^*/\overset{*}{\leftrightarrow}_T$. The *word problem for a Thue system* T on Δ^* is then the following:

Given two words $u, v \in \Delta^*$, do we have $u \overset{*}{\leftrightarrow}_T v$? This can be rephrased as : are u and v in the same equivalence class in $\Delta^*/\overset{*}{\leftrightarrow}_T$?

Let now Σ be a finite alphabet, and $\theta \subseteq \Sigma \times \Sigma$ be a binary reflexive and symmetric relation on Σ. Whenever $(a, b) \in \theta$, each occurrence of ab (resp. ba) in any word $w \in \Sigma^*$ can be replaced by ba (resp. ab). If a word $v \in \Sigma^*$ is derived from a word $w \in \Sigma^*$ by such a sequence of replacements, then we denote it by $v \equiv_\theta w$. It is clear that $\equiv_\theta$ as defined is an equivalence relation. Taking the quotient $\Sigma^*/\equiv_\theta$, we obtain the so-called *free partially commutative monoid generated by* Σ *with respect to the concurrency relation* θ.

The *word problem for free partially commutative monoids* is the following:

Given two words $v, w \in \Sigma^*$, is $v \equiv_\theta w$? Relying on a result of [5], it has been shown [4] that, for fixed Σ, the word problem in free partially commutative monoids is decidable in linear time in the length of the inputs v and w.

3 Abisha, Thomas and Subramanian's Monoid-Based System

The FPCM *system.* In this part we briefly recall the public key system based on free partially commutative monoids [1].

Public Key. An alphabet Δ, a Thue system $T \subseteq \Delta^* \times \Delta^*$ and $y_0, y_1 \in \Delta^*$.

Secret Key. An alphabet Σ of cardinality smaller than that of Δ, a concurrency relation $\theta \subseteq \Sigma \times \Sigma$ and $g \in \mathrm{Hom}(\Delta^*, \Sigma^*)$ - the set of monoid homomorphisms between Δ^* and Σ^*, mapping one letter to one letter or to the empty word[3] - such that, for each $(l, r) \in T$, we have either:

$$\big(g(l), g(r)\big) \in \big\{(ab, ba), (ba, ab)\big\}, \text{ for some } (a, b) \in \theta, \text{ or } g(l) = g(r). \tag{1}$$

Therefore, $u \overset{*}{\leftrightarrow}_T v$ implies $g(u) \equiv_\theta g(v)$, for any $u, v \in \Delta^*$.

Additionally, two words x_0, $x_1 \in \Sigma^*$ such that $x_0 \not\equiv_\theta x_1$, $x_0 \equiv_\theta g(y_0)$ and $x_1 \equiv_\theta g(y_1)$ are also kept secret. Thus the secret key is $(\Sigma, \theta, g, x_0, x_1)$.

Encryption. To encrypt a bit $b \in \{0, 1\}$, Bob repeatedly apply to y_b rewriting rules specified by the Thue system T. The word $c \in \Delta^*$ resulting from this process is then sent to Alice.

[3] In the sequel, the term morphism will always denote an application that maps one letter to one letter or to the empty word.

Decryption. To decrypt a ciphertext $c \in \Delta^*$, Alice first computes $g(c) \in \Sigma^*$ and then solves the (easy) word problem in the free partially commutative monoid generated by θ. If $g(c) \equiv_\theta x_0$, then the plaintext is 0, otherwise it is 1. We refer to [1] for further details concerning this system.

The underlying hard problem: TMMI
Let Δ be an alphabet and $T \subseteq \Delta^* \times \Delta^*$ be a Thue system. The hard problem underlying the above system is the one that we call *Thue Monoid Morphism Interpretation* (TMMI) problem in the sequel. The essence of the problem is that of recovering an alphabet $\tilde{\Sigma}$, a concurrency relation $\tilde{\theta} \subseteq \tilde{\Sigma} \times \tilde{\Sigma}$ and a nontrivial (so-called) *interpretation morphism* $\tilde{g} \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}^*)$ such that

$$\forall u, v \in \Delta^*,\ u \overset{*}{\longleftrightarrow}_T v \implies \tilde{g}(u) \equiv_{\tilde{\theta}} \tilde{g}(v).$$

This problem was introduced by Abisha, Thomas, and Subramanian in [1] and has been proven to be NP-HARD. A more precise statement of it will be given in section 5.

Remark: it has to be emphasized that the morphism g chosen for the FPCM system satisfies a slightly weaker property than a general interpretation morphism as defined above does. Indeed, as expression (1) shows, the constraint on g is only to map rules of T onto congruent words modulo θ *of the special form* (ab, ba) or (ba, ab), for $(a, b) \in \theta$, and not onto any congruent words modulo θ. This restriction will be exploited in the next section.

4 Cryptanalysis of the FPCM System

Our purpose is here to break the instances of the TMMI problem that are used in FPCM systems. Before giving our algorithm, we shall briefly present the idea. Our concern is to find a triple - say $(\tilde{\Sigma}, \tilde{\theta}, \tilde{g})$ - that will play the same role as the secret key (Σ, θ, g), i.e. that "fits" the public key (Δ, T) in the sense of expression (1). First, notice that concerning the alphabet Σ, the only thing that matters is its cardinality. Indeed, any alphabet $\tilde{\Sigma}$ of size $|\Sigma|$ will yield a solution which is isomorphic to (Σ, θ, g). Next, observe that to find an equivalent secret key, it is sufficient to find a suitable $\tilde{g}$; the concurrency relation $\tilde{\theta}$ will then follow from $\tilde{g}$.

Thus, we shall focus on the retrieval of a (suitable) $\tilde{g}$.

To do this, we shall - for each possible cardinality i, $1 \leq i \leq |\Delta|$, of some alphabet $\tilde{\Sigma}$, and for every morphism $\tilde{g}$ from Δ^* onto $\tilde{\Sigma}^*$ - execute the following algorithm. We know that a solution exists for $i = |\Sigma|$; besides, a design specification of the FPCM system is that $|\Sigma| << |\Delta|$. Thus we know that a solution will be found for i much smaller than $|\Delta|$.

In this algorithm, $\tilde{\Sigma}_i = \{\tilde{\sigma}_1, \cdots, \tilde{\sigma}_i\}$ is an alphabet of $i \geq 1$ symbols. For each $i, 1 \leq i \leq \Delta$ and each $\tilde{g}_i \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)$, we run:

Algorithm 1
Input: $\tilde{g}_i \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)$ and a Thue system $T \subseteq \Delta^* \times \Delta^*$.
Output: A concurrency relation $\tilde{\theta}_i \subseteq \tilde{\Sigma}_i \times \tilde{\Sigma}_i$ or `Error`.
$\tilde{\theta}_i \leftarrow \emptyset$
For $(u, v) \in T$ **do**
  Compute $\tilde{g}_i(u)$ and $\tilde{g}_i(v)$
  If $\tilde{g}_i(u) \neq \tilde{g}_i(v)$ **then**
    If there exist $\tilde{\sigma} \neq \tilde{\sigma}' \in \tilde{\Sigma}_i$, s.t. $\tilde{g}_i(u) = \tilde{\sigma}\tilde{\sigma}'$ and $\tilde{g}_i(v) = \tilde{\sigma}'\tilde{\sigma}$ **then**
      If $(\tilde{\sigma}, \tilde{\sigma}')$ or $(\tilde{\sigma}', \tilde{\sigma})$ are not in $\tilde{\theta}_i$ **then** $\tilde{\theta}_i \leftarrow \tilde{\theta}_i \cup (\tilde{\sigma}, \tilde{\sigma}')$
    Else `Error`
  EndIf
EndFor
Return $\tilde{\theta}_i$

4.1 Proof of Correctness of the Algorithm

Proposition 1. *Let T be a Thue system on Δ generated as in section 3. Then:*

i) There exists an index $i \geq 1$ and $\tilde{g} \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)$ for which algorithm 1 returns a concurrency relation $\tilde{\theta} \subseteq \tilde{\Sigma}_i \times \tilde{\Sigma}_i$ such that $(\tilde{\Sigma}_i, \tilde{\theta}, \tilde{g})$ is an equivalent secret key for (Δ, T).
ii) $x_0 \not\equiv_\theta x_1 \Rightarrow \tilde{g}(y_0) \not\equiv_{\tilde{\theta}} \tilde{g}(y_1)$.
iii) The pair $(\tilde{g}, \tilde{\theta})$ is recovered by performing $O\big(|T||\Sigma|(|\Sigma|+1)^{|\Delta|}\big)$ operations.

Proof. i). Let k be the number of symbols of Σ and $\pi \in \mathrm{Isom}(\Sigma^*, \tilde{\Sigma}_k^*)$ (the set of monoid isomorphisms from Σ^* onto $\tilde{\Sigma}_k^*$, $\tilde{\Sigma}_k$ being any alphabet of size k). By construction, we know that there exists a concurrency relation $\theta \subseteq \Sigma \times \Sigma$ and an interpretation morphism g for the Thue system T. From these, we define $\tilde{g} = \pi \circ g \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_k^*)$ and $\tilde{\theta} = \big\{(\pi(a), \pi(b)) : \exists (u, v) \in T$, with

$$(g(u), g(v)) \in \{(ab, ba), (ba, ab)\}, \text{ for some } (a, b) \in \theta\big\}.$$

It is easy to see that, on input $\tilde{g} \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_k^*)$, algorithm 1 returns $\tilde{\theta}$.

Let $(u, v) \in T$. To conclude the proof of *i)*, we remark that we have the following equivalences:

$\big(\big(g(u), g(v)\big) \in \big\{(ab, ba), (ba, ab)\big\}$, for some $(a, b) \in \theta\big) \iff \big(\big(\tilde{g}(u), \tilde{g}(v)\big) \in \big\{(\tilde{a}\tilde{b}, \tilde{b}\tilde{a}), (\tilde{b}\tilde{a}, \tilde{a}\tilde{b})\big\}$, for some $(\tilde{a}, \tilde{b}) \in \tilde{\theta}\big)$, and $\tilde{g}(u) = \tilde{g}(v) \iff g(u) = g(v)$.

Therefore for each $(u, v) \in T$, we have either: $\big(\tilde{g}(u), \tilde{g}(v)\big) \in \big\{(\tilde{a}\tilde{b}, \tilde{b}\tilde{a}), (\tilde{b}\tilde{a}, \tilde{a}\tilde{b})\big\}$ for some $(\tilde{a}, \tilde{b}) \in \tilde{\theta}$, or $\tilde{g}(u) = \tilde{g}(v)$, meaning that $(\tilde{\Sigma}_k, \tilde{\theta}, \tilde{g})$ is an equivalent secret key for (Δ, T).

ii). We define $\pi(\theta) = \big\{(\pi(a), \pi(b)) : (a, b) \in \theta\big\}$. One can see at once that, as $x_0 = g(y_0)$ and $x_1 = g(y_1)$, we have $x_0 \not\equiv_\theta x_1 \iff \tilde{g}(y_0) \not\equiv_{\pi(\theta)} \tilde{g}(y_1)$. Therefore, as $\tilde{\theta} \subseteq \pi(\theta)$, the fact that $x_0 \not\equiv_\theta x_1$ implies $\tilde{g}(y_0) \not\equiv_{\tilde{\theta}} \tilde{g}(y_1)$.
iii). We first remark that $\tilde{\theta}$ is immediately deduced from the knowledge of $\tilde{g}$.

Indeed, from $\tilde{g}$, $\tilde{\theta}$ is simply defined as

$$\tilde{\theta} = \big\{(\tilde{\sigma}, \tilde{\sigma}') \in \tilde{\Sigma}_k \times \tilde{\Sigma}_k,\ \exists (u,v) \in T,\ \text{with } (\tilde{g}(u), \tilde{g}(v)) \in \{(\tilde{\sigma}\tilde{\sigma}', \tilde{\sigma}'\tilde{\sigma}), (\tilde{\sigma}'\tilde{\sigma}, \tilde{\sigma}\tilde{\sigma}')\}\big\}.$$

Therefore, the complexity of recovering the pair $(\tilde{g}, \tilde{\theta})$ is then equal to the one of recovering $\tilde{g}$. To estimate this complexity, we simply have to enumerate the number of elements in $\mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)$, for each $i, 1 \leq i \leq k$. Since each $g_i \in \mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)$ is uniquely determined by the images of the symbols of Δ on $\tilde{\Sigma}_i \cup \{\lambda\}$, we deduce that $|\mathrm{Hom}(\Delta^*, \tilde{\Sigma}_i^*)| = (i+1)^{|\Delta|}$. Finally, $\tilde{g}$ is obtained by performing at most $\sum_{i=1}^{k} (i+1)^{|\Delta|}$ iterations of the algorithm, and the pair $(\tilde{g}, \tilde{\theta})$ is recovered with $O\big(|T|k(k+1)^{|\Delta|}\big)$ operations, where an operation here means an evaluation of $\tilde{g}$.

Now let (Δ, T, y_0, y_1) be a public key of the FPCM system. In light of property 1-*i*), we can recover from this key a triple $(\tilde{\Sigma}_k, \tilde{\theta}, \tilde{g})$, such that for every two words u and v in Δ^* with $u \overset{*}{\longleftrightarrow}_T v$, we have $\tilde{g}_i(u) \equiv_{\tilde{\theta}_i} \tilde{g}_i(v)$. Moreover, according to *ii*), setting $\tilde{x}_0 = \tilde{g}(y_0)$ and $\tilde{x}_1 = \tilde{g}(y_1)$ permit to decrypt all ciphertexts $c \in \Delta^*$ generated from (Δ, T, y_0, y_1), in the same way as x_0 and x_1 of Σ^* do : indeed, given a ciphertext c, we simply compute its image through $\tilde{g}$, and if $\tilde{g}(c) \equiv_{\tilde{\theta}} \tilde{x}_0$ then the cleartext is 0, otherwise it is 1.

Practical Complexity: in practice, the complexity of the above method is much better than the one given in *iii)* of property 1. Indeed, let (Σ, θ, g) be a solution of the TMMI problem, and set $k = |\Sigma|$. Let also $\tilde{\Sigma}_k$ be an alphabet of size k. It is to be noted that any isomorphism π from Σ^* onto $\tilde{\Sigma}_k^*$ will also yield a solution, namely $(\tilde{\Sigma}_k, \pi(\theta), \pi \circ g)$, with $\pi(\theta) = \{(\pi(a), \pi(b)),\ (a,b) \in \theta\}$. Thus, as $|\mathrm{Isom}(\Sigma^*, \tilde{\Sigma}_k^*)| = k!$, the expected number of iterations of the algorithm is $\big(\sum_{i=1}^{k} (i+1)^{|\Delta|}\big)/k!$, and consequently the expected number of operations to find a solution by this method is

$$O\left(\frac{\big(|T|k(k+1)^{|\Delta|}\big)}{k!}\right).$$

As an illustration, on the FPCM system given as an example in [1] (with $\Delta = 9$, $|T| = 11$, $|\Sigma| = 3$), the bound of proposition 1 yields $O(2^{23})$, whereas the expected complexity above gives $O(2^{18})$.

The bounds given here permit to have an idea of the parameter sizes to choose in order to reach a reasonable level of security (indeed, in [1], no hint was given on the suitable parameter sizes). For instance, with $|\Delta| = 28$, $|T| \approx |\Delta|$ and $|\Sigma| = |\Delta|/2$, the bound above yields $O(2^{80})$. Note that $|\Delta| = 28$ is yet likely to result in quite impractical schemes in terms of expansion rate and encryption cost. Moreover, it remains unclear how to derive a secure public key/secret key pair of satisfactory size (in the sense of the above bound).

5 Breaking the TMMI Problem

We shall now give an algorithm that breaks the hard problem underlying both the FPCM system and the zero-knowledge protocol on monoids proposed in [1]. First, let us state the TMMI problem and its hardness result, as in [1].

Theorem 1. *Given a Thue system T on an alphabet Δ and two words y_0 and y_1, the problem of constructing a nontrivial interpretation morphism $g : \Delta^* \to \Sigma^*$, so that g maps T into a free partially commutative monoid on Σ with a concurrency relation θ is NP-hard. Here, g maps the words y_0 and y_1 respectively to the words x_0 and x_1, with $x_0 \not\equiv_\theta x_1$, g satisfying the following conditions:*
- for each letter $d \in \Delta$, $g(d)$ is either a letter in Σ or the empty word λ.
- there exists a letter $d \in \Delta$ such that $g(d)$ is a letter in Σ.
- for every two words u and v in Δ^* with $u \overset{*}{\longleftrightarrow}_T v$, we have $g(u) \equiv_\theta g(v)$.
- $g(y_0) = x_0$, $g(y_1) = x_1$.

Our algorithm relies heavily on the following result of Cori and Perrin, which is indeed the core of the easiness of the word problem in free partially commutative monoids.

In what follows, Σ is as usual a finite alphabet, and $\theta \subseteq \Sigma \times \Sigma$ is a concurrency relation on Σ. For $B \subseteq \Sigma$, let π_B denote the monoid morphism from Σ^* onto B^*, defined by $\pi_B(b) = b$ if $b \in B$, $\pi_B(b) = \lambda$ if $b \in \Sigma \setminus B$.

Proposition 2. *[5] Let $u, v \in \Sigma^*$. We have $u \equiv_\theta v$ if, and only if:*

$$(i) \quad \pi_{\{a\}}(u) = \pi_{\{a\}}(v),\ \forall a \in \Sigma, \text{ and}$$
$$(ii) \quad \pi_{\{a,b\}}(u) = \pi_{\{a,b\}}(v),\ \forall (a,b) \notin \theta.$$

The proof can be found in [5].

5.1 The Algorithm

For each possible cardinality i of some alphabet Σ_i, and for every morphism g_i from Δ^* onto Σ_i^*, we execute algorithm 2 until a solution is found or until[4] $i = |\Delta|$. Indeed, according to condition 1 of theorem 1, an interpretation morphism would map Δ onto an alphabet $g(\Delta)$ of size at most that of Δ; thus it suffices to consider alphabets Σ of size at most $|\Delta|$.

The algorithm constructs a concurrency relation - say θ_i - fitting T and g_i. To construct θ_i, we make use of the "if" part of the previous proposition on the images of the rules of T by g_i: as they have to be congruent modulo a concurrency relation θ - provided a solution (Σ, θ, g) exists for (Δ, T) - we force condition (ii) of proposition 2 to be true on those images (having checked beforehand that condition (i) of proposition 2 is fulfilled).

Note that if the algorithm returns `Error`, it means that g_i is not a suitable interpretation morphism; thus the algorithm stops and has to be rerun with

[4] Actually, the case $i = |\Delta|$ is not worth considering, as we shall explain in the paragraph concerning the proof of correctness of algorithm 2.

another g_i. If, for a given i, no suitable g_i has been found, we increment i and rerun the algorithm. The index i is here in the integer range $[1, |\Delta|]$.

Algorithm 2
Input: $g_i \in \mathrm{Hom}(\Delta^*, \Sigma_i^*)$, a Thue system $T \subseteq \Delta^* \times \Delta^*$, y_0 and $y_1 \in \Delta^*$.
Output: A concurrency relation $\theta_i \subseteq \Sigma_i \times \Sigma_i$ or `Error`.

$\theta_i \leftarrow \emptyset$
For $(u, v) \in T$ **do**
  Compute $g_i(u)$ and $g_i(v)$
  For $a \in \Sigma_i$ **do**
    If $\pi_{\{a\}}(g_i(u)) \neq \pi_{\{a\}}(g_i(v))$ **then** `Error`
  EndFor
  For $(a, b) \in \Sigma_i \times \Sigma_i$ **do**
    If $\pi_{\{a,b\}}(g_i(u)) \neq \pi_{\{a,b\}}(g_i(v))$
      If (a, b) or (b, a) are not in θ_i **then** $\theta_i \leftarrow \theta_i \cup \{(a, b)\}$
  EndFor
EndFor
If $g_i(y_0) \equiv_{\theta_i} g_i(y_1)$ **then** `Error`
Return θ_i

5.2 Proof of Correctness of the Algorithm

Theorem 2. *Let T be a Thue system on an alphabet Δ. Then*
i) If, for each i, $1 \leq i \leq |\Delta|$, and for each $g_i \in \mathrm{Hom}(\Delta^*, \Sigma_i^*)$, algorithm 2 outputs "`Error`", then there is no solution to the TMMI problem for (Δ, T).
ii) If there exists an i for which, on input $(\Delta, T, \Sigma_i, g_i, y_0, y_1)$, algorithm 2 returns a concurrency relation θ_i, then $(\Sigma_i, \theta_i, g_i)$ is a solution of the TMMI problem for the instance (Δ, T, y_0, y_1).

Proof. i). If a solution (Σ, θ, g) of TMMI for (Δ, T) were to exist, every rule of T should be mapped to a pair of words that are congruent modulo θ. If, for one letter $a \in \Sigma_i$, we reach the "`Error`" state of the loop on T for a pair $(u, v) \in T$, it means that condition (i) of proposition 2 is not fulfilled, and thus there is no concurrency relation making $g_i(u)$ be congruent to $g_i(v)$ for this g_i. Thus, g_i is not a good candidate, and we must try with another g_i.

On the other hand, if we reach the "`Error`" state after the loop on T, then $g_i(y_0) \equiv_{\theta_i} g_i(y_1)$. Thus θ_i is not a good candidate, and we also must try with another g_i.

As said above, an interpretation morphism would map Δ onto an alphabet of size at most that of Δ. As the algorithm performs an exhaustive search on all morphisms from Δ onto an alphabet of size i, for all possible i's from 1 to $|\Delta|$, it follows that if no morphism satisfying the conditions of proposition 2 on T is found, it means that there exists no interpretation morphism compatible with (Δ, T).

ii). Let θ_i be the concurrency relation output by the algorithm, on input $(\Delta, T, \Sigma_i, g_i, y_0, y_1)$. To prove that $(\Sigma_i, \theta_i, g_i)$ is a solution of the TMMI problem

for (Δ, T, y_0, y_1), it is sufficient to show that $g_i(y_0) \not\equiv_{\theta_i} g_i(y_1)$ - which is indeed trivially satisfied by the very construction of g_i - and that

$$u \overset{*}{\leftrightarrow}_T v \implies g_i(u) \equiv_{\theta_i} g_i(v). \tag{2}$$

To show (2), first note that this implication is equivalent to

$$(u, v) \in T \implies g_i(u) \equiv_{\theta_i} g_i(v). \tag{3}$$

Indeed, (2) $\Rightarrow$ (3) is straightforward. The converse follows from the transitivity properties of $\overset{*}{\leftrightarrow}_T$ and $\equiv_{\theta_i}$: indeed, suppose (3) is true. Let $u, v \in \Delta^*$, with $u \overset{*}{\leftrightarrow}_T v$. Then there exists a finite sequence $u_0, u_1, \ldots, u_k$, $k \in \mathbb{N}^*$, of words of Δ^* such that

$$u_0 = u \leftrightarrow_T u_1 \leftrightarrow_T u_2 \leftrightarrow_T \ldots \leftrightarrow_T u_{k-1} \leftrightarrow_T u_k = v.$$

For $0 \leq j \leq k-1$, $u_j \leftrightarrow_T u_{j+1}$ means that there exists $(\ell, r) \in T$, and $x, y \in \Delta^*$, with $u_j = x\ell y$ and $u_{j+1} = xry$. This yields $g_i(u_j) = g_i(x)g_i(\ell)g_i(y)$ and $g_i(u_{j+1}) = g_i(x)g_i(r)g_i(y)$. As (3) holds, we have $g(\ell) \equiv_{\theta_i} g(r)$, so that $g_i(u_j) \equiv_{\theta_i} g_i(u_{j+1})$. Thus, we have the corresponding sequence

$$g_i(u) \equiv_{\theta_i} g_i(u_1) \equiv_{\theta_i} g_i(u_2) \equiv_{\theta_i} \ldots \equiv_{\theta_i} g_i(u_{k-1}) \equiv_{\theta_i} g_i(v),$$

so that (2) is true.

Let now $(u, v) \in T$. According to algorithm 2, θ_i is constructed such that, whenever $(a, b) \notin \theta_i$, then $\pi_{\{a,b\}}(g_i(u)) = \pi_{\{a,b\}}(g_i(v))$. This is exactly condition (ii) of proposition 2. Besides, condition (i) of this proposition is satisfied for $g_i(u)$ and $g_i(v)$, otherwise the algorithm would not have returned any concurrency relation. Thus, again by this proposition, we obtain that $g_i(u) \equiv_{\theta_i} g_i(v)$.

Note on the case $i = \Delta$:

If a solution (Σ, θ, g) of TMMI for (Δ, T, y_0, y_1) were to exist for $|\Sigma| = |\Delta|$, g is either an isomorphism, in which case g maps any concurrency relation to a concurrency relation, or $(\tilde{\Sigma}, \tilde{\theta}, g)$ is a solution for (Δ, T, y_0, y_1), with $\tilde{\Sigma} = \{g(\delta) : \delta \in \Delta\}$ and $\tilde{\theta} = \{(g(\sigma), g(\sigma')) \in \tilde{\Sigma} \times \tilde{\Sigma} : (\sigma, \sigma') \in \theta\}$. Since g is not an isomorphism, we then have that $\tilde{\Sigma} \subsetneq \Sigma$, so that this case must have been handled at a former stage of the algorithm.

5.3 Complexity Considerations

One iteration of algorithm 2 involves $O(|\Delta|^2|T|)$ operations, where here an operation is an evaluation of π or of g_i. In the worst case (no solution, or a solution for $|\Sigma| = |\Delta| - 1$), algorithm 2 will be iterated $\sum_{i=1}^{|\Delta|-1} |\mathrm{Hom}(\Delta, \Sigma_i)|$ times.

With $|\mathrm{Hom}(\Delta, \Sigma_i)| = (i+1)^{|\Delta|}$, and using the rough bound $(i+1)^{|\Delta|} \leq |\Delta|^{|\Delta|}$, for all $1 \leq i \leq |\Delta| - 1$, we get that the number of iterations of algorithm 2 is[5] $O\big(|\Delta|^{|\Delta|+1}\big)$. Thus, to solve[6] TMMI one needs to perform $O\big(|T||\Delta|^{|\Delta|+3}\big)$ operations.

[5] In all cases, i.e. on instances admitting a solution or not.

[6] To find a solution or conclude that there is no.

Practical Complexity: For instances of the TMMI problem admitting a solution - say (Σ, θ, g), with $k = |\Sigma|$ - note that, by construction, g must be such that $\forall (u,v) \in T,\ \pi_{\{a\}}(g(u)) = \pi_{\{a\}}(g(v)), \forall a \in \Sigma$, and $g(y_0) \not\equiv_{\theta_i} g_i(y_1)$. Therefore, any isomorphism $\pi \in \mathrm{Isom}(\Sigma, \Sigma)$, will also yield a solution, namely $(\Sigma, \pi(\theta), \pi \circ g)$, with $\pi(\theta) = \{(\pi(a), \pi(b)),\ (a,b) \in \theta\}$. Thus the expected number of iterations of the algorithm is $\left(\sum_{i=1}^{k} (i+1)^{|\Delta|}\right)/k!$, and the expected number of operations to find a solution by this method is $O\left(\frac{C_k}{k!}\right)$, where C_k is the number of[7] operations on instances admitting a solution with an alphabet of size k, that is $C_k = O\big(|T|\, k^{|\Delta|+3}\big)$.

6 The Group Setting

In [1], Abisha, Thomas and Subramanian have proposed a group variant of the FPCM system, that we called FPCG system at the beginning of this paper. Before describing it, we need

A few additional notations. Let Δ be a finite alphabet. We denote by $\Delta^{-1} = \{a^{-1},\ a \in \Delta\}$, a set of formal inverses for the letters in Δ, and we set $\Delta^{\pm} = \Delta \cup \Delta^{-1}$. Let R be a finite subset of $\Delta^{\pm *} \times \{\lambda\}$. We let $\leftrightarrow_R$ be the binary symmetric relation defined for any $x, y \in \Delta^{\pm *}$ by $x \leftrightarrow_R y$ if, and only if, one of the following cases hold:
$x = urv$ and $y = uv$, with $(r, \lambda) \in R$, and $u, v, \in \Delta^{\pm *}$,
$x = ua^{\epsilon}a^{-\epsilon}v$ and $y = uv$, for $a \in \Delta$, $\epsilon = \pm 1$, and $u, v, \in \Delta^{\pm *}$,

As before, we denote by $\overset{*}{\leftrightarrow}_R$ the reflexive transitive closure of $\leftrightarrow_R$. A so-called *finitely presented group* G is then the quotient $\Delta^{\pm *}/\overset{*}{\leftrightarrow}_R$, that we equivalently denote by $G =< \Delta, R >$. The set R is the set of *defining relations* of G. We shall denote $x \equiv_R y$ whenever two words x and y are in the same equivalence class in G. Note that this amounts to saying that $xy^{-1} \equiv_R \lambda$.

The word problem for the group G is that of deciding, for any two words $x, y \in \Delta^{\pm *}$, whether $x \equiv_R y$, or equivalently whether $xy^{-1} \equiv_R \lambda$ holds.

This problem has been proven undecidable for general G [7].

Let Σ be a finite alphabet, and let θ_0 be a partial commutativity relation (concurrency relation) on Σ. We denote by θ, the extension of θ_0 to $\Sigma^{\pm}$, i.e. $\theta = \{(a,b), (a^{-1},b), (a,b^{-1}), (a^{-1},b^{-1}),\ (a,b) \in \theta_0\}$. We shall write the four pairs $(a,b), (a^{-1},b), (a,b^{-1}), (a^{-1},b^{-1})$ in a concise way as $(a^{\pm}, b^{\pm})$, for $a, b \in \Sigma$. With this notation, we have $\theta = \{(a^{\pm}, b^{\pm}),\ (a,b) \in \theta_0\}$. Let

$$R(\theta) = \{(cd(dc)^{-1}, \lambda),\ (c,d) \in \theta\} \subset \Sigma^{\pm *} \times \{\lambda\}.$$

Then $R(\theta)$ is a presentation of the *free partially commutative group* $G(\theta) \overset{\text{def}}{=} \Sigma^{\pm *}/\overset{*}{\leftrightarrow}_{R(\theta)}$. The word problem for free partially commutative groups has been proven decidable in linear time [16].

[7] Actually this is an upper bound on the number of operations, as evaluated above, but in the case of instances admitting a solution.

6.1 The FPCG System

We here follow the exposition of [1]. Let Σ, θ_0, θ, $R(\theta)$ and $G(\theta)$ be defined as above. Let also x_1 be a word of $\Sigma^{\pm *}$, such that $x_1 \not\equiv_{R(\theta)} \lambda$. Denote by Δ, an alphabet of cardinality much larger than that of Σ, and by g, a group morphism from $\Delta^{\pm *}$ to $\Sigma^{\pm}$. Let y_0 and y_1 be two words of $\Delta^{\pm *}$ such that $g(y_0) \equiv_{R(\theta)} \lambda$ and $g(y_1) \equiv_{R(\theta)} x_1$.

Additionally, a finitely presented group $G =< \Delta, \bar{R} >$ is constructed from a subset $\bar{R}$ of $\Delta^{\pm *} \times \{\lambda\}$ satisfying, for all $(uv^{-1}, \lambda) \in \bar{R}$, one of the following conditions:

(c1) $(g(u)g(v)^{-1}, \lambda) \in R(\theta)$
(c2) $g(u) \equiv_{R(\theta)} \lambda$ and $g(v) \equiv_{R(\theta)} \lambda$.

The public key is then (G, y_0, y_1), and the secret key is $(G(\theta), x_1, g)$.

Encryption of bit 0 is done by choosing any word $w \in \Delta^{\pm *}$ equivalent to y_0 w.r.t. $\bar{R}$. Decryption of w is done by first computing $g(w)$ and solving the easy word problem in $G(\theta)$ to check if $g(w) \equiv_{R(\theta)} \lambda$.

Note that θ_0 completely determines $G(\theta)$. Thus, to break this system, it suffices to recover a triple $(\tilde{\Sigma}, \tilde{\theta}, \tilde{g})$, where $\tilde{g}$ is a group morphism from $\Delta^{\pm *}$ to $\tilde{\Sigma}^{\pm *}$, and $\tilde{\theta}$ is a concurrency relation on the alphabet $\tilde{\Sigma}$, $\tilde{g}$ and $\tilde{\theta}$ being compatible with the Thue system $\bar{R}$ in the sense of conditions $(c1)$ and $(c2)$.

6.2 Breaking the FPCG System

Although FPCG and FPCM systems are quite alike, one cannot directly run algorithm 1 for breaking FPCG. Indeed, while condition (c1) of FPCG is the exact transcript of condition (1) of section 3, condition (c2) is not handled by algorithm 1. In other words, attempting to construct a suitable concurrency relation, say θ_i, fitting $\bar{R}$ and a morphism g_i, algorithm 1 as it is will not add the pairs in θ_i corresponding to those rules $(uv^{-1}, \lambda) \in \bar{R}$ that map to pairs $(g_i(u), g_i(v))$ satisfying condition (c2) with $(g, R(\theta))$ being replaced by $(g_i, R(\theta_i))$, with $R(\theta_i) = \{(cd(dc)^{-1}, \lambda), (c, d) \in \theta_i\} \subseteq \Sigma_i^{\pm *} \times \{\lambda\}$. We shall call S_2, the subset of $\bar{R}$ of such rules.

Remark: of course, in the case when condition (c2) is simply reduced to $g(uv^{-1}) = \lambda$, for all rules in S_2, algorithm 1 works with no modification. Another case when algorithm 1 works as it stands is when the concurrency relation obtained by this algorithm makes all rules $(uv^{-1}, \lambda) \in \bar{R}$ with $(g(uv^{-1}), \lambda) \notin R(\theta_i)$ satisfy $g(u) \equiv_{R(\theta_i)} \lambda$ and $g(v) \equiv_{R(\theta_i)} \lambda$.

Handling condition (c1):
Although algorithm 1 permits to fill θ_i with relations enabling the rules $(uv^{-1}, \lambda) \in \bar{R} \setminus S_2$ to be mapped to rules of $\theta_i$test on the morphism g_i. On the other hand, property (i) of proposition 2 does the job in the monoid case. In order to use a similar property, we derive an analog of the Cori-Perrin test in the group setting (see proposition 3 below). But contrary to the monoid case, one cannot obtain a pure analogue to this test. Indeed, the necessary condition is straightforward, but due to simplifications by trivial relations of the form (aa^{-1}, λ), the projections π are no more length preserving, and thus one looses the sufficient condition.

As before, let Σ be a finite alphabet. For $B \subseteq \Sigma$, let $\pi_{B^\pm}$ denote the group morphism from ${\Sigma^\pm}^*$ to ${B^\pm}^*$, defined by $\pi_{B^\pm}(b) = b$ if $b \in B^\pm$, $\pi_{B^\pm}(b) = \lambda$ if $b \in \Sigma^\pm \setminus B^\pm$. Using the above definition of $R(\theta)$, we get:

Proposition 3. *Let $u, v \in {\Sigma^\pm}^*$. If $u \equiv_{R(\theta)} v$ then*

$$(i)\quad \pi_{\{a,a^{-1}\}}(u) = \pi_{\{a,a^{-1}\}}(v),\ \forall a \in \Sigma,\ \textit{and}$$
$$(ii)\quad \pi_{\{a,a^{-1},b,b^{-1}\}}(u) = \pi_{\{a,a^{-1},b,b^{-1}\}}(v),\ \forall (a,b) \notin \theta$$

hold.

Proof. Let $u, v \in {\Sigma^\pm}^*$, with $u \equiv_{R(\theta)} v$. Then it is plain that (i) holds. Moreover, the words u and v being congruent modulo $R(\theta)$, it is also clear that their projections onto pairs (a, b) not belonging to θ must yield equal words in ${\Sigma^\pm}^*$, as those cannot be modified by the relations of $R(\theta)$. In other words, condition (ii) of the proposition is satisfied.

With this proposition, one can, in the same spirit as in algorithm 2 for TMMI, construct a concurrency relation using property (ii) above. Furthermore, the resulting relation contains the one obtained by algorithm 1: indeed, by property (ii), one actually also adds in θ_i all pairs (c, d) for which $\pi_{\{c,c^{-1},d,d^{-1}\}}(g_i(uv^{-1})) = cd(dc)^{-1}$, and thus all the images of rules of $\bar{R}$ that are rules of $R(\theta)$. Thus, instead of algorithm 1, we perform a loop on $\bar{R}$ as of algorithm 2 in order to construct a concurrency relation fitting condition (c1). This will be the first part of our algorithm to break FPCG, and we shall call $\tilde{\theta}_i$ the relation obtained in this way. For the sake of clarity, we shall now denote $\equiv_{R(\theta_i)}$ simply by $\equiv_{\theta_i}$.

Handling condition (c2):
Note that, concerning rules of $\bar{R}$, one only knows uv^{-1}, and thus one only has to deal with[8] $g_i(uv^{-1})$. But we can observe that, for $(uv^{-1}, \lambda) \in \bar{R}$, we have:

$$(uv^{-1}, \lambda) \in S_2 \Leftrightarrow \big(g_i(u) \equiv_{\theta_i} \lambda \text{ and } g_i(v) \equiv_{\theta_i} \lambda\big) \Rightarrow g_i(uv^{-1}) \equiv_{\theta_i} \lambda.$$

We shall call $E_{\bar{R}}$, the subset of $\bar{R}$ of rules for which $g_i(uv^{-1}) \not\equiv_{\theta_i} \lambda$, i.e. $E_{\bar{R}} = \bar{R} \cap \{(z, \lambda),\ z \in \Delta^{\pm *},\ g_i(z) \not\equiv_{\theta_i} \lambda\}$. Thus we have, for $(uv^{-1}, \lambda) \in \bar{R}$:

$$(uv^{-1}, \lambda) \in E_{\bar{R}} \Rightarrow (uv^{-1}, \lambda) \notin S_2.$$

To begin with, we set $\theta_i = \tilde{\theta}_i$. For a word w, we denote by $\mathrm{alp}(w)$ the set of letters of w, and $\mathrm{alp} = \cup_{(uv^{-1},\lambda) \in E_{\bar{R}}} \mathrm{alp}(g_i(uv^{-1}))$. Also, $(\underline{a}, \underline{b})$ denotes an ℓ-tuple of pairs of letters $((a_1, b_1), \ldots, (a_\ell, b_\ell))$, for an integer $\ell \in \mathbb{N}^*$.

The second part of our algorithm will perform a loop on all possible ℓ-tuples of pairs of letters of $\Sigma_i \times \mathrm{alp}$, until if finds one ℓ-tuple $(\underline{a}, \underline{b})$ that will make all the $g_i(uv^{-1})$ - for $(uv^{-1}, \lambda) \in E_{\bar{R}}$ - be congruent to λ modulo the relation $\theta_i \cup \{(a_i, b_i),\ (a_i, b_i) \in (\underline{a}, \underline{b})\}$. In this case, it returns this augmented relation that we still call θ_i. If no such ℓ-tuple satisfies this property, then we increment

[8] and not with $g_i(u)$ and $g_i(v)$.

ℓ and restart this loop. We do so until ℓ reaches a predefined bound, say ℓ'. Note that ℓ represents the number of rules of $\bar{R}$ that actually are in S_2. Thus, ℓ' is at worse equal to $|\bar{R}|$, but in practice, we can expect that no more than $|\bar{R}|/2$ rules of $\bar{R}$ are in S_2, and thus $\ell' \leq |\bar{R}|/2$ in practice[9].

If no relation has been found for $\ell \leq \ell'$, then we restart the algorithm with another g_i. If no relation is found for any g_i, then we increment i and restart the whole process. At the end of the algorithm, if no `Error` is returned, we come up with a relation θ_i such that the rules of $\bar{R}$ are mapped by g_i onto words of Σ_i^* satisfying either $g_i(uv^{-1}) = cd(dc)^{-1}$, with $(c,d) \in \tilde{\theta}_i \subseteq \theta_i$(i.e. cond. (c1)), or $g_i(uv^{-1}) \equiv_{\theta_i} \lambda$. We now give the algorithm, as well as a proof that it indeed breaks FPCG (prop. 4).

Algorithm 3
Input: $g_i \in \mathrm{Hom}(\Delta^{\pm *}, \Sigma_i^{\pm *})$, $\bar{R} \subseteq \Delta^{\pm *} \times \Delta^{\pm *}$, $y_0, y_1 \in \Delta^{\pm *}$ and ℓ'.
Output: A concurrency relation $\theta_i \subseteq \Sigma_i^{\pm} \times \Sigma_i^{\pm}$ or `Error`.

$\tilde{\theta}_i \leftarrow \emptyset$
For $(uv^{-1}, \lambda) \in \bar{R}$ **do**
  Compute $g_i(uv^{-1})$
  For $a \in \Sigma_i$ **do**
    If $\pi_{\{a,a^{-1}\}}(g_i(uv^{-1})) \neq 0$ **then** `Error`
  For $(a,b) \in \Sigma_i \times \Sigma_i$ **do**
    If $\pi_{\{a,a^{-1},b,b^{-1}\}}(g_i(uv^{-1})) \neq 0$ **then** $\tilde{\theta}_i \leftarrow \tilde{\theta}_i \cup \{(a^{\pm}, b^{\pm})\}$
EndFor
If $g_i(y_0) \equiv_{\tilde{\theta}_i} g_i(y_1)$ **then** `Error`
$\theta_i \leftarrow \tilde{\theta}_i$
$E_{\bar{R}} = \bar{R} \cap \{(z, \lambda),\ z \in \Delta^{\pm *},\ g(z) \not\equiv_{\theta_i} \lambda\}$
$\mathrm{alp} \leftarrow \cup_{(uv^{-1},\lambda) \in E_{\bar{R}}} \mathrm{alp}(g_i(uv^{-1}))$
$\ell \leftarrow 1$
While $\ell \leq \ell'$ **do**
  For $(\underline{a}, \underline{b}) \in ({\Sigma_i}^{\pm} \times \mathrm{alp})^{\ell}$ **do**
    If $g_i(uv^{-1}) \equiv_{\theta_i \cup (\cup_{j=1}^{\ell} \{({a_j}^{\pm}, {b_j}^{\pm})\})} \lambda$ for all $(uv^{-1}, \lambda) \in E_{\bar{R}}$ **then**
      $\theta_i \leftarrow \theta_i \cup \left(\cup_{j=1}^{\ell} \{({a_j}^{\pm}, {b_j}^{\pm})\} \right)$
      If $g_i(y_0) \not\equiv_{\theta_i} g_i(y_1)$ **then Return** θ_i
  EndFor
  $\ell \leftarrow \ell + 1$
EndWhile
Return `Error`

Proposition 4. *Let $(G = (\Delta, \bar{R}), y_0, y_1)$ be the public key of an* FPCG *system, generated as in section 6.1, and let $(G(\theta) = (\Sigma, \theta), x_1, g)$ be the corresponding secret key. Set $k = |\Sigma|$. Then*

[9] In the example given in [1], no rule of $\bar{R}$ maps to condition (c2) by g, i.e. $S_2 = \emptyset$.

i) There exists an index $i \geq 1$ *and* $g_i \in \mathrm{Hom}(\Delta^{\pm *}, \Sigma_i^{\pm *})$ *for which algorithm* 3 *returns a concurrency relation* $\theta_i \subseteq \Sigma_i^{\pm *} \times \Sigma_i^{\pm *}$ *such that* $(\Sigma_i, \theta_i, g_i)$ *permits to decrypt all ciphertexts of this system.*
ii) $x_0 \not\equiv_\theta x_1 \Rightarrow g_i(y_0) \not\equiv_{\theta_i} g_i(y_1)$.
iii) The pair (g_i, θ_i) *is recovered by performing* $O\big(k(k+1)^{|\Delta|}(|\bar{R}|(k^2 + \ell' k^{2\ell'}))\big)$ *operations*[10].

Proof. i). We know that there exists a triple (Σ, θ_0, g) such that, for all $(uv^{-1}, \lambda) \in \bar{R}$, $g(uv^{-1}) \in \Sigma^{\pm *}$ satisfies either condition (c1) or (c2). Let then $k = |\Sigma|$.

Let $\psi \in \mathrm{Isom}(\Sigma^{\pm *}, \Sigma_k^{\pm *})$ (the set of group isomorphisms between $\Sigma^{\pm *}$ and $\Sigma_k^{\pm *}$). Define $g_k = \psi \circ g \in \mathrm{Hom}(\Delta^{\pm *}, \Sigma_k^{\pm *})$ and

$$\tilde{\theta}_{0,k} = \big\{(\psi(a), \psi(b)) : \exists (uv^{-1}, \lambda) \in \bar{R}, g(uv^{-1}) = ab(ba)^{-1} \text{ for some } (a,b) \in \theta_0\big\}.$$

Then, on input g_k, the loop on $\bar{R}$ of algorithm 3 returns $\tilde{\theta}_k$, such that:

$$\{(c^\pm, d^\pm),\ (c,d) \in \tilde{\theta}_{0,k}\} \subseteq \tilde{\theta}_k.$$

Thus, $\tilde{\theta}_k$ contains all pairs $(c,d) \in \Sigma_k^\pm \times \Sigma_k^\pm$, for which there exists $(uv^{-1}, \lambda) \in \bar{R}$, such that $g_k(uv^{-1}) = cd(dc)^{-1}$.

Let θ_k by the relation obtained from $\tilde{\theta_k}$ at the end of the loop over ℓ. Then, by the very construction of this loop, the algorithm will exactly add to $\tilde{\theta}_k$ all the pairs of $\Sigma_k^\pm \times \Sigma_k^\pm$ needed for the images by g_k of the remaining rules of $\bar{R}$ - i.e. those whose images under g_k are not of the form $= cd(dc)^{-1}$ nor congruent to the empty word modulo $\tilde{\theta}_k$ - to be congruent to the empty word modulo θ_k. Thus the concurrency relation obtained at the end of the algorithm is such that, for all rules (uv^{-1}, λ) of $\bar{R}$, we have either $(g_k(uv^{-1}), \lambda) \in R(\theta_k)$ or $g_k(uv^{-1}) \equiv_{\theta_k} \lambda$. Consequently, since *ii*) is plainly true, the triple $(\Sigma_k, \theta_k, g_k)$ enables to decrypt all ciphertexts encrypted with $(\Delta, \bar{R})$, in the same way as (Σ, θ, g) does.

iii). The complexity of the algorithm can be evaluated as follows:

The loop on $\bar{R}$ will take $O\big(|\bar{R}|k^2\big)$ operations. For each ℓ, the loop on $\Sigma_i \times \mathrm{alp}$ needs $O\big(|\bar{R}|k^{2\ell}\big)$ op. Thus, for ℓ' iterations of this loop, we get a complexity of $O\big(\ell'|\bar{R}k^{2\ell'}\big)$ op. Counting the number of group morphisms from $\Delta^{\pm *}$ to $\Sigma_i^{\pm *}$, we have, for each i, $O\big((i+1)^{|\Delta|}(|\bar{R}|(k^2 + \ell' k^{2\ell'}))\big)$ op. Thus, the overall complexity of the method to find a suitable (g_i, θ_i) is $O\big(k(k+1)^{|\Delta|}(|\bar{R}|k^2 + \ell' k^{2\ell'})\big)$ op.

7 Conclusion

In light of our cryptanalyses of FPCM and FPCG systems, and having in mind the chosen ciphertext attack of [8], the use of those schemes as they are seems hard to achieve. Concerning the zero-knowledge protocol based on the underlying hard problem TMMI [1], our attack of section 5 also shows that it is insecure. The group-based version of this protocol can still be of interest, as the group variant of TMMI has not been broken yet. Indeed, one further line of research would here be to find an attack on this hard problem.

[10] An operation being an evaluation of g_i or of π.

References

1. P.J. Abisha, D. G. Thomas, K. G. Subramanian. *Public Key Cryptosystems Based on Free Partially Commutative Monoids and Groups.* Proceedings of INDOCRYPT 2003, LNCS 2904, Springer, pp.218-227.
2. A. Aho, R. Sethi, J. Ullman. *Code optimization and finite Church-Rosser systems.* Design and Optimization of Computers, R. Rustin Ed., Prentice-Hall 1972, pp.89-105.
3. R.V. Book. *Confluent and other types of Thue systems.* Journal of the ACM 29, 1982, pp.171-182.
4. R.V. Book, H.N. Liu. *Rewriting systems and word problems in a free partially commutative monoid.* Information Processing Letters 26, 1987/88, pp.29-32.
5. R. Cori, D. Perrin. *Automates et commutations partielles.* R.A.I.R.O. Informatique théorique 19, 1985, pp.21-32.
6. J. Kari. *A cryptanalytic observation concerning systems based on language theory.* Discrete Applied Mathematics 21, 1988, pp.45-53.
7. P. S. Novikov. *On the algorithmic unsolvability of the word problem in group theory.* Trudy Mat. Inst. Steklov 44 (1955), pp.1-143.
8. M.I. González-Vasco, R. Steinwandt. *Pitfalls in public key systems based on free partially commutative monoids an groups.* Cryptology ePrint archive 2004/012.
9. M.I. González-Vasco, R. Steinwandt. *A Reaction Attack on a Public Key Cryptosystem Based on the Word Problem.* Applicable Algebra Engineering, Communication and Computing, 14(5), 2004, pp.335-340.
10. V. A. Oleshchuk *On Public-Key Cryptosystem Based on Church-Rosser String-Rewriting Systems.* Extended Abstract. Proceedings of COCOON'95, LNCS 959, Springer-Verlag, pp.264-269.
11. A. Salomaa. *A public key cryptosystem based on language theory.* Computers and Security 7, 1988, pp.83-87.
12. R. Siromoney, L. Matthew. *A public key cryptosystem based on Lyndon words.* Information Processing Letters 35, 1990, pp.33-36.
13. K.G. Subramanian, R. Siromoney, P.J. Abisha. *A DOL-TOL public key cryptosystem.* Information Processing Letters 26, 1987, pp.95-97.
14. A.M. Turing *The word problem in semi-groups with cancellation.* Annals of Math.(2) vol.52, 1950, pp.491-505.
15. N. R. Wagner, M. R. Magyarik. *A public key cryptosystem based on the word problem.* Proceedings of CRYPTO'84, LNCS 96, Springer-Verlag, pp.19-36.
16. C. Wrathall. *The word problem for free partially commutative groups.* Journal of Symbolic Computation vol 6., 1988, pp.99-104.

Exact Analysis of Montgomery Multiplication

Hisayoshi Sato[1], Daniel Schepers[2,⋆], and Tsuyoshi Takagi[2]

[1] Hitachi, Ltd., Systems Development Laboratory,
292, Yoshida-cho, Totsuka-ku, Yokohama, 244-0817, Japan
hisato@sdl.hitachi.co.jp
[2] Technische Universität Darmstadt, Fachbereich Informatik,
Hochschulstr.10, D-64289 Darmstadt, Germany
{schepers, takagi}@informatik.tu-darmstadt.de

Abstract. The Montgomery multiplication is often used for efficient implementations of public-key cryptosystems. This algorithm occasionally needs an extra subtraction in the final step, and the correlation of these subtractions can be considered as an invariant of the algorithm. Some side channel attacks on cryptosystems using Montgomery Multiplication has been proposed applying the correlation estimated heuristically. In this paper, we theoretically analyze the properties of the final subtraction in Montgomery multiplication. We investigate the distribution of the outputs of multiplications in the fixed length interval included between 0 and the underlying modulus. Integrating these distributions, we present some proofs with a reasonable assumption for the appearance ratio of the final subtraction, which have been heuristically estimated by previous papers. Moreover, we present a new invariant of the final subtraction: $x \cdot y$ with $y = 3x \bmod m$, where m is the underlying modulus. Finally we show a possible attack on elliptic curve cryptosystems using this invariant.

Keywords: timing attack, elliptic curve cryptosystem, Montgomery multiplication, randomization.

1 Introduction

The Montgomery Multiplication is widely utilized in implementations for public-key cryptosystems [10]. The Montgomery multiplication is an efficient algorithm for computing modular multiplication without the use of relatively expensive division with remainder, and it is suitable for the memory-constraint devices such as smart cards. In the past years researchers like Dhem, Quisquater or Schindler [4, 11] etc. attacked this operation. Nevertheless it is still used in hardware implementations and can be used for attacks.

Since 1996 timing attacks gained more and more interest. After Kocher [7, 8] started with the first attacks on DSS and RSA numerous researchers worked on

⋆ The second author is supported by SicAri Project (www.sicari.de) — German Federal Ministry of Education and Research.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 290–304, 2004.

this topic. RSA and DES were probably the targets which have been attacked most. This kind of attack is especially attractive to smart cards. Dhem et al. proposed the first timing attack on RSA using Montgomery multiplication [4]. They focused on the final subtraction which appears in the Montgomery multiplication. They experimentally showed a timing attack by analyzing the distribution of the appearance ratio correlated to the secret information. From their experiment the appearance ratio is about 17% on average. Walter [18] proposed a Montgomery multiplication without final subtraction, and Hachez and Quisquater improved it [17]. However, these schemes cause overhead costs comparing with the original Montgomery multiplication.

After the timing attack, some theoretical analysis about the final subtraction have been investigated. Schindler heuristically showed a relationship between the appearance ratio and the underlying parameters [11]. He estimated the appearance ratio is $\frac{x \bmod m}{2R}$, where $x \in \mathbb{Z}/n\mathbb{Z}$ and R is the Montgomery constant. On the other hand, Walter and Thomson estimated that the ratio for a squaring is 0.33 and that for a multiplication is 0.25 if the modulus m is near to Montgomery constant R [16]. The attacker is able to distinguish a squaring and a multiplication by observing the final subtraction of Montgomery multiplication.

In this paper, we present some exact analysis on Montgomery multiplication under a reasonable assumption. Firstly we divide the interval between 0 and the underlying modulus into intervals with length R, then we investigate the distribution of outputs of multiplications in each interval. Integrating these results, we prove that the appearance ratios of the final subtraction in Montgomery multiplication and squaring are asymptotically $\frac{m}{4R}$ and $\frac{m}{3R}$, respectively. The assumption effects only the case that $m \approx R$ where m is the modulus and R is the Montgomery constant. Schindler's heuristic function $\frac{x \bmod m}{2R}$ is proved as well. This assumption describes clearly the behavior of the Montgomery multiplication's final subtractions.

We present a new variant of the final subtraction attack as well. Namely we show that the multiplication $x \cdot y$ with $y = 3x \bmod m$ has a different subtraction ratio from both multiplication and squaring. This operation often appears in the addition formula of elliptic curve cryptosystems. We show a possible timing attack based on this invariant. Indeed, the randomization presented by Coron's 3rd [3] could be vulnerable to the attack. This is different to the attack of Goubin [5] because we have the opportunity of choosing more points than a special one of the curve. Finally we show an experimental result on the appearance ratio discussed in this paper.

This paper is organized as follows: In Section 2 we shortly review the Montgomery multiplication and the timing attack using the final subtraction of the Montgomery multiplication. In Section 3 we present the proposed exact estimation about the appearance ratio of the final subtraction. In Section 4 we show a new timing attack and its analysis. In Section 5 we state the concluding remark.

2 Montgomery Multiplication and Timing Attack

In this section we shortly review the Montgomery multiplication and some timing attacks using the appearance probability of the last subtraction.

2.1 Montgomery Multiplication

The Montgomery Multiplication [10] is an efficient algorithm for computing modular multiplications without using relatively expensive divisions, and is widely utilized for public-key cryptosystems. Especially, it is suitable for the memory-constraint devices such as smart cards.

Note that the Montgomery multiplication has outputs slightly different from ordinary modular multiplications. In an exponentiation these can be corrected by three extra Montgomery multiplication. Because the Montgomery multiplication outputs results in the residue class without any divisions it is the fastest way to multiply. This is because if the radix b is chosen suitably the divisions are only shifts. Shifts are basic operations in hardware and are therefore fast. The following algorithm is taken from [9]

Algorithm 1: Montgomery multiplication

Input: $m = (m_{n-1} \cdots m_0)_b,\ X = (x_{n-1} \cdots x_0)_b,\ Y = (y_{n-1} \cdots\ y_0)_b,\ b = 2^k,$
$R = b^n,\ \gcd(m, b) = 1,\ m' = -m^{-1} \bmod b.$
Output: $XYR^{-1} \bmod m$
1. $A \leftarrow 0$ $(A = (a_n \cdots a_0)_b)$.
2. For i from 0 to $(n-1)$ do:
  $temp \leftarrow 0$,
  For j from 0 to $(n-1)$ do:
    $\{temp, a_j\} \leftarrow x_j y_i + a_j + temp$,
  $a_n \leftarrow temp,\ temp \leftarrow 0,\ u_i \leftarrow a_0 m' \bmod b$,
  For j from 0 to n do:
    $\{temp, a_j\} \leftarrow m_j u_i + a_j + temp$,
  $A \leftarrow A/b$.
3. If $A \geq m$, $A \leftarrow A - m$. $\Leftarrow$ **Final Subtraction**
4. Return(A).

The running time of the steps can be analyzed as follows: The computations in step 2 are expected to take approximately constant time for fixed n. This is because of the repetition in every multiplication and the constant n repetitions of the for-loops. After step 2 the value of A varies between 0 and twice the modulus. A subtraction has to be done if A is larger than the modulus. This subtraction is called *final subtraction*.

2.2 Timing Attack and Its Analysis

We shortly review the timing attack on RSA cryptosystem using the Montgomery multiplication.

Dhem et al. simulated a timing attack on the CASCADE smart card [4]. They focused that the probability of the final subtraction depends on the message and the secret bit. The attacker can guess the secret bit by observing the distribution of the final subtraction. The authors stated the final subtraction occurs in a multiplication of two random inputs in about 17% of the time. They expected a 512-bit RSA key to be cracked within a few minutes once 350 000 timing measurements are collected. Later on Koeune, Quisquater and Schindler [6] improved the result and cracked a 512-bit RSA key with 5000 timings by success rate 0.5.

There are some theoretical estimations for the probability of the final subtraction. Walter and Thomson investigated the probability of the final subtraction appeared in Montgomery multiplication and further on Walter and Schindler improved the results [16, 15, 13]. Finally Schindler improved the results once again. But these accurate analysis was not provable for us. [12]

They showed the following estimations under several convenient conditions for simplicity.

$$P_{mul} = \frac{R}{4m}\left(1-\left(1-\frac{m}{R}\right)^2\right) - \left(1-\frac{m}{R}\right) - \frac{R}{2m}\left(1-\frac{m}{R}\right)^2 \log\left(1-\frac{m}{R}\right), \quad (1)$$

$$P_{sqr} = 1 - \frac{2R}{3m}\left(1-\left(1-\frac{m}{R}\right)^{3/2}\right), \quad (2)$$

where P_{mul}, P_{sqr} are the probability of the final subtraction appeared in Montgomery multiplication for general multiplications and squarings respectively. Interestingly, the probability for squaring is $1/3$ and that for multiplication is $1/4$ for $m \approx R$. It is an open problem to show a general formula of the probability.

Schindler proposed another timing attack on RSA using the Chinese remainder theorem [11]. He estimated heuristically the probability of the final subtraction is

$$\frac{c \bmod m}{2R}, \quad (3)$$

where c is a cipher-text and m is the secret modulus. The secret modulus m can be calculated by the chosen cipher-text setting. As he stated in the paper, the precise proof for the formula is not given yet.

3 Exact Analysis of Montgomery Multiplication

In this section we analyze the distribution of the final subtraction in Montgomery multiplication. We will investigate the distribution for the general case and some special cases, and summarize these in section 3.5.

In case of $R = b$ $(n = 1)$, Montgomery Multiplication is given by the following simple form:

ALGORITHM 2: MONTGOMERY MULTIPLICATION - SPECIAL CASE

Input: $m,\ X,\ Y,\ R$, $\gcd(m, R) = 1,\ m' = -m^{-1} \bmod R$
Output: $XYR^{-1} \bmod m$
S-1. $u \leftarrow xym' \bmod R$.
S-2. $A \leftarrow (xy + um)/R$.
S-3. If $A \geq m$, $A \leftarrow A - m$.
S-4. Return(A).

First of all, we will reduce the problem for ALGORITHM 1 to that for ALGORITHM 2. Thus we will prove the following lemma.

Lemma 1. *For same inputs m, X, Y and R of* ALGORITHM *1 and* ALGORITHM *2, the final subtraction in step 3 of* ALGORITHM *1 is performed if and only if the final subtraction in step S-3 of* ALGORITHM *2 is performed.*

Proof. In step 2 of ALGORITHM 1, in order to distinguish, let us denote A for each i by A_i. Then it can be easily seen that

$$A_{n-1} = \frac{xy + (\sum_{i=0}^{n-1} u_i)m}{b^n},$$

and we can see that the subtraction in step 3 is performed if and only if $A_{n-1} \geq m$. Let us set $S = \sum_{i=0}^{n-1} u_i$. Then by the validity of Montgomery Multiplication, we have that A_{n-1} is an integer, namely, $xy + Sm \equiv 0 \bmod b^n$. Hence $S \equiv -xy/m \bmod R$. Moreover, as an integer, $S < R$, thus we have $S = (-xy/m \bmod R)$.

Note that the right hand side is an integer not less than 0 and less than R. Therefore, we have that the subtraction in step 3 is performed if and only if $xy + (-xy/m \bmod R)m \geq mR$, and this condition is nothing less than the equivalent condition for the final subtraction in step S-3 of ALGORITHM 2. □

3.1 Preparation

In the following, we will consider the problem for ALGORITHM 2. After step S-2 we obtain the following equation:

$$A = (xy + (xym' \bmod R)m)/R \tag{4}$$

Thus we can see that

$$A \geq m \ \Leftrightarrow\ xy + (xym' \bmod R)m \geq mR. \tag{5}$$

$$g(m, R) := \#\left\{w \in \mathbb{Z} \mid 0 \leq w \leq (m-1)^2,\ w + (wm' \bmod R)m \geq mR\right\}. \tag{6}$$

When we represent $w = \eta + \xi R$, $0 \leq \eta < R$, $0 \leq \xi \leq (m-1)^2/R$, then the equation in the right side of (6) becomes

$$\eta + \xi R + (\eta m' \bmod R)m. \tag{7}$$

This number should be divisible by R, so that we can represent $(\eta m' \bmod R)m = -\eta + \pi R$ for some integer $\pi = \pi(\eta)$ depending on η. Moreover, we know $\pi \le (R-1)(m+1)/R$ due to $0 \le (\eta m' \bmod R)m \le (R-1)m$. Therefore, if $m < R-1$ holds, then we obtain $0 \le \pi \le m-1$. Next, we assume the following distribution.

Assumption *DIS.* $\alpha := \eta m' \bmod R$ distributes in interval $0 \le \alpha < R$ uniformly and randomly for R-fold different η.

We know that this assumption is adequate experientially[†]. From this assumption, we can see that π distributes in interval $0 \le \pi < m$ uniformly and randomly for R-fold different η. Indeed, for $0 < \eta, \eta' < m$, it is easy to see that $\pi(\eta) = \pi(\eta')$ if and only if $\eta = \eta'$. Moreover the random distribution of $\eta m' \bmod R$ induces the random distribution of π. Hence we can see that one π corresponds to $R/(m+1)$-fold η on average, namely there is an $(R/(m+1))$-to-1 map between π and η. On the other hand, Equation (7) can be represented as $R(\xi + \pi)$. If $\xi + \pi \ge m$, then $R(\xi + \pi)$ is greater than mR. For fixed ξ, the conditions in (6) is true with ξ-fold π that satisfies $m - \xi \le \pi \le m-1$. We know that ξ satisfies $0 \le \xi \le (m-1)^2/R$, and thus we have obtained

$$g(m,R) \;\approx\; \sum_{\xi=0}^{\frac{(m-1)^2}{R}} \frac{R}{m+1}\xi \;\approx\; \int_1^{\frac{(m-1)^2}{R}} \frac{Rx}{m+1}dx \;\approx\; \frac{m^3}{2R} - \frac{R}{2m}. \tag{8}$$

Note that we used $m \pm 1 \approx m$ for the final approximation.

3.2 Distribution of the Final Subtraction in the General Case

Next, we consider the distribution of $xyR^{-1} \bmod m$ with the final subtraction in the following. Previously we set $w = xy$, but xy is not uniformly distributed in interval $[0,(m-1)^2]$ for $0 \le x, y \le m-1$. We consider the divided interval $[0,(m-1)^2]$ with width R. In general, we set $G_N := \{0,1,2,\cdots,N-1\} \subset \mathbb{Z}$ for natural integer N and let $\phi = \phi_N$ be the multiplication map:

$$\phi : G_N \times G_N \to G_{(N-1)^2+1}, \quad (x,y) \mapsto xy.$$

For fixed ξ, the value $w = \eta + \xi R$ runs between ξR and $(\xi+1)R$. Denote by $F_\phi(\xi)$ the number of the images of $\phi_m : F_\phi(\xi) := \#\{\ \mathrm{Im}(\phi_m) \cap [\xi R, (\xi+1)R)\ \}$. Then, for fixed ξ the probability that the integers in $[\xi R, (\xi+1)R)$ are equal to the image of map ϕ_m is given by $F_\phi(\xi)/R$. In the words, the number of π that are contained in the image of ϕ_m is given by

$$\frac{F_\phi(\xi)}{R}\xi. \tag{9}$$

[†] Since $m' \in (\mathbb{Z}/R\mathbb{Z})^\times$, the m'-multiplication map $\eta \mapsto \eta m' \bmod R$ is bijective. Thus $\eta m' \bmod R$ are uniformly distributed.

On the other hand, let $G_\phi(\xi)$ denote the number of integers $0 \le x, y \le m-1$ whose images by $\phi = \phi_m$ are in the interval $[\xi R, (\xi+1)R)$:

$$G_\phi(\xi) := \#\{(x,y) \in G_m \times G_m \mid \phi_m(x,y) \in [\xi R, (\xi+1)R)\}.$$

From $\xi R \le xy \le (\xi+1)R$ and the condition of x, y, we have

$$\frac{\xi R}{m} \le x < m, \tag{10}$$

and for a fixed x, the number of y that satisfies the conditions is exactly R/x (more precisely we should consider its floor value). Hence we have

$$G_\phi(\xi) \approx \sum_{\xi R/m \le x < m} \frac{R}{x} \approx R(2\log m - \log R - \log \xi).$$

Therefore, among the image of ϕ_m from the interval $[\xi R, (\xi+1)R)$, there are $G_\phi(\xi)/F_\phi(\xi) \approx R(2\log m - \log R - \log \xi)/F_\phi(\xi)$ elements mapped from (x,y) on average. Consequently, for fixed ξ, the number of images of the map ϕ_m is equal to $F_\phi(\xi)\xi/R$ among ξ-fold π. Each image has $R(2\log m - \log R - \log \xi)/F_\phi(\xi)$-fold pre-images of (x,y) on average. Therefore, for w in $[\xi R, (\xi+1)R)$, the number that satisfies (4) with x, y is

$$\frac{R(2\log m - \log R - \log \xi)}{F_\phi(\xi)} \cdot \frac{F_\phi(\xi)}{R}\xi \cdot \frac{R}{m+1} = \frac{R(2\log m - \log R - \log \xi)\xi}{m+1}.$$

Let $s(m,R)$ denote the number of (x,y) that satisfies Equation (5):

$$s(m,R) := \#\left\{(x,y) \in \mathbb{Z} \times \mathbb{Z} \mid 0 \le x, y \le m-1,\ xy + (xym' \bmod R)m \ge mR\right\}.$$

Then from the above argument, we have the following approximation formula.

$$\begin{aligned}
s(m,R) &\approx \frac{R}{m+1} \sum_{\xi=1}^{(m-1)^2/R} \left(\log \frac{m^2}{R} - \log \xi\right)\xi \\
&\approx \frac{R}{m+1} \int_1^{(m-1)^2/R} \left(\log \frac{m^2}{R} - \log x\right) x dx \qquad (11) \\
&\approx \frac{R}{m+1} \left\{\frac{1}{4}\left(\frac{(m-1)^2}{R}\right)^2 + \left(1 - \log \frac{m^2}{R}\right)\right\} \qquad (12) \\
&\approx \frac{m^3}{4R} + \frac{R}{m}\left(1 - \log \frac{m^2}{R}\right). \qquad (13)
\end{aligned}$$

Here, the transformation from (11) to (13) is obtained by the partial derivation and $m \pm 1 \approx m$.

3.3 The Case of $x = y$

We consider the case of $x = y$, which simplifies $t(m, R)$ that satisfies equation (5). Thus we will estimate the following.

$$t(m,R) := \#\left\{x \in \mathbb{Z} \mid 0 \le x \le m-1,\ x^2 + (x^2 m' \bmod R)m \ge mR\right\}.$$

We follow the estimation for the general case. Let $G_\psi(\xi)$ denote the number of integers $0 \le x \le m-1$ whose images by $\psi(x) = \psi_m(x) := x^2$ are in the interval $[\xi R, (\xi+1)R)$:

$$G_\psi(\xi) := \#\{x \in G_m \mid \psi_m(x) \in [\xi R, (\xi+1)R)\}.$$

Because of $\sqrt{\xi R} \le x < \sqrt{(\xi+1)R} < m$, we have

$$G_\psi(\xi) \ \approx \sum_{\sqrt{\xi R} \le x < \sqrt{(\xi+1)R}} 1 \ \approx \ \sqrt{(\xi+1)R} - \sqrt{\xi R}.$$

Hence, among the image of ψ_m in the interval $[\xi R, (\xi+1)R)$, there are $G_\psi(\xi)/F_\psi(\xi) \approx (\sqrt{(\xi+1)R} - \sqrt{\xi R})/F_\psi(\xi)$ elements mapped from x on average, where $F_\psi(\xi)$ denote the number of the images of ψ_m: $F_\psi(\xi) := \{\ \mathrm{Im}(\psi_m)\ \cap \#[\xi R, (\xi+1)R)\ \}$. Therefore, for w in $[\xi R, (\xi+1)R)$, the number that stratifies (5) with x is

$$\frac{\sqrt{(\xi+1)R} - \sqrt{\xi R}}{F_\psi(\xi)} \cdot \frac{F_\psi(\xi)}{R}\xi \cdot \frac{R}{m+1} = \frac{\sqrt{R}}{m+1}\left(\sqrt{(\xi+1)} - \sqrt{\xi}\right)\xi.$$

Thus we have following approximation.

$$\begin{aligned} t(m,R) &\approx \frac{\sqrt{R}}{m+1} \sum_{\xi=1}^{\frac{(m-1)^2}{R}} \left(\sqrt{\xi+1} - \sqrt{\xi}\right)\xi \\ &\approx \frac{\sqrt{R}}{m+1} \int_1^{\frac{(m-1)^2}{R}} \left(\sqrt{x+1} - \sqrt{x}\right) x dx \\ &\approx \frac{\sqrt{R}}{m+1}\left(\frac{1}{3}\left(\frac{(m-1)^2}{R}\right)^{3/2} + \frac{15}{8}\left(\frac{(m-1)^2}{R}\right)^{1/2}\right). \end{aligned}$$

As in the previous section, using $m \pm 1 \approx m$ and ignoring small constant, we have

$$t(m,R) \approx \frac{m^2}{3R}. \tag{14}$$

3.4 The Case of Fixed x

We consider the case that x is fixed in the following. Let x be an integer such that $0 \le x < m$, and fix. If the multiplication xy for $0 \le y < m$ lies in the interval $[\xi R, (\xi+1)R)$, then from the equation (10), we have

$$\xi \le \frac{mx}{R}. \tag{15}$$

In this case, for R/x-folds y, the image of ϕ_m is in $[\xi R, (\xi+1)R)$ (if $\xi > mx/R$, then no image for y is in this interval). On the other hand, we have to consider the distribution of $xym' \bmod R$ for m-fold y instead of that of $\eta m' \bmod R$ for R-fold η in Assumption *DIS*, and the former strongly depends on the fixed x. We will focus on the gcd of x and R in the following.

Lemma 2. *Let $x' = \gcd(x, R)$. Then for any $r(< R)$, there exists some $s = s(r) < R/x'$ such that $xr \bmod R = sx'$ $(< R$ as an integer).*

Proof. As an integer, let $xr = \alpha R + \beta$, $\beta \leq R - 1$, then we have $\beta \equiv 0 \bmod x'$. Hence putting $\beta = sx'$ as an integer, we have $s \leq (R-1)/x'$ and $xr \bmod R = sx'$. □

Using this lemma, in the equation (4), there exists $s \leq (R/x') - 1$ such that $xym' \bmod R = x's$. Hence we have $xy + (xym' \bmod R)m = xy + x'sm \leq mR + xy - x'm$. Therefore, for y such that $y < x'm/x$, the subtraction is not performed. So from equation $\xi m/x < y$, for ξ satisfying

$$\xi < \frac{x'm}{R}, \tag{16}$$

the subtraction is not performed. Hence, similarly to the general case, an approximation of the number

$$u(x, m, R) := \#\{y \in \mathbb{Z} \mid 0 \leq y \leq m - 1,\ xy + (xym' \bmod R)m \geq mR\}$$

is given by following (using $m \pm 1 \approx m$).

$$u(x, m, R) \ \approx\ \frac{R}{x(m+1)} \sum_{\xi = \frac{m\gcd(x,R)}{R}}^{\frac{mx}{R}} \xi \ \approx\ \frac{m}{2xR}(x^2 - \gcd(x, R)^2). \tag{17}$$

Remark 1. In case of $z := -x/m \bmod R$ is very small (e.g. $z = 1, 2, \ldots$) or very large(e.g. $z = R-1, R-2, \ldots$), we can see that there are some bias. In order to explain these bias, we need to consider u as a function of x, m, R and z.

3.5 Comparison of Probability

There are m^2 inputs for the general case and m inputs for the case of $x = y$, $y = ax \bmod m$ and fixed x. Therefore, from the previous sections, we have obtained the following probabilities.

$$\frac{g(m, R)}{m^2} \approx \frac{m}{2R}, \qquad \frac{s(m, R)}{m^2} \approx \frac{m}{4R}, \qquad \frac{t(m, R)}{m} \approx \frac{m}{3R},$$

$$\frac{u(x, m, R)}{m} \ \approx\ \frac{1}{2xR}(x^2 - \gcd(x, R)^2).$$

Consequently, we obtain the following theorem.

Theorem 1. *We assume that the assumption DIS is true. The final subtraction of Montgomery multiplication asymptotically appears with probability* $\frac{m}{4R}$*. If two inputs are equal (i.e. Montgomery squaring), then the probability becomes* $\frac{m}{3R}$*.*

If we choose $m \to R$, then these ratios for Montgomery multiplication and squaring converge $\frac{1}{4}$ and $\frac{1}{3}$, respectively. On the other hand, for $m \to R/2$, these ratios converge $\frac{1}{8}$ and $\frac{1}{6}$, respectively.

Corollary 1. *For randomly chosen* m*, the average ratio of the final subtraction in Montgomery multiplication (or Montgomery squaring) is asymptotically about 0.188 (or 0.250), respectively.*

Proof. From the assumption, m randomly distributes in interval $[\frac{R}{2}, R]$. Then the average ratio for Montgomery multiplication is $\frac{3}{16} = 0.1875$. Similarly, we can estimate $\frac{1}{4} = 0.25$ for Montgomery squaring. □

4 Application to Elliptic Curve Cryptosystems

In this section we shortly review elliptic curve cryptosystems, and side channel attack on them. Then we show a new invariant of a special Montgomery multiplication used for elliptic curve cryptosystems.

4.1 Elliptic Curve Cryptosystems

Elliptic curves over a finite prime field $K = GF(m)$ with $m > 3$ are defined by

$$E : \{(x, y) \in K^2 | y = x^3 + ax + b\} \cup \{\mathcal{O}\}, \tag{18}$$

where $a, b \in K,\ 4a^3 + 27b^2 \neq 0$, and $\mathcal{O}$ is a point at infinity. The Elliptic curve E has a group structure with neutral element $\mathcal{O}$. The group operation of the elliptic curve is as follows:

Let E denote an elliptic curve and $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ denote points on the curve then $-P_1 = (x_1, -y_1)$. $P_3 = P_1 + P_2$ is calculated by

$$x_3 = \begin{cases} \lambda_1^2 - x_1 - x_2 & : \quad P_1 \neq P_2 \\ \lambda_2^2 - 2x_1 & : \quad P_1 = P_2 \end{cases} \qquad y_3 = \begin{cases} (x_1 - x_3)\lambda_1 - y_1 & : \quad P_1 \neq P_2 \\ (x_1 - x_3)\lambda_2 - y_1 & : \quad P_1 = P_2 \end{cases}$$

where $\lambda_1 = \frac{y_1 - y_2}{x_1 - x_2}$ and $\lambda_2 = \frac{3x_1^2 + a}{2y_1}$.

We denote by ECADD by the first formula and ECDBL by the second, respectively. In order to avoid the expensive inversion operation in the affine coordinates, we usually use the *Jacobian* coordinates [2]. A point $P = (x, y)$ in the affine coordinates is represented by $P = (X, Y, Z)$ with $x = X/Z^2$ and $y = Y/Z^3$ in the Jacobian coordinates. The addition formula in the Jacobian coordinates is as follows:

ECDBL in Jacobian Coordinates (ECDBL$^{\mathcal{J}}$)**:**
$X_3 = T$, $Y_3 = -8{Y_1}^4 + M(S - T)$, $Z_3 = 2Y_1Z_1$,
$S = 4X_1{Y_1}^2$, $M = 3{X_1}^2 + a{Z_1}^4$, $T = -2S + M^2$.

ECADD in Jacobian Coordinates (ECADD$^{\mathcal{J}}$)**:**
$X_3 = -H^3 - 2U_1H^2 + R^2$, $Y_3 = -S_1H^3 + R(U_1H^2 - X_3)$, $Z_3 = Z_1Z_2H$,
$U_1 = X_1{Z_2}^2$, $U_2 = X_2{Z_1}^2$, $S_1 = Y_1{Z_2}^3$, $S_2 = Y_2{Z_1}^3$, $H = U_2 - U_1$, $R = S_2 - S_1$.

The group offers the scalar multiplication of $k \cdot P, k \leq ord(E)$ for a point P with order q on a curve E. A standard double-and-add algorithm can compute the scalar multiplication, but it is not secure against the timing attack. The double-and-add-always method can resist the timing attack [3].

Algorithm 3: Coron dummy

Input: $d = (d_{n-1} \cdots d_1 d_0)_2$, $P \in E(K)$ $(d_{n-1} = 1)$
Output: dP.
1. $Q[0] \leftarrow P$
2. For $i = (n-2)$ down to 0 do:
   $Q[0] \leftarrow ECDBL(Q[0])$,
   $Q[1] \leftarrow ECADD(Q[0], P)$
   $Q[0] \leftarrow Q[d_i]$
3. Return$(Q[0])$.

4.2 DPA and Countermeasure

The differential power analysis (DPA) observes many power consumptions and analyze these information together with statistic tools. Even if a method is secure against the timing attack, it might not be secure against the DPA. The DPA attacker tries to guess that the computation cP for an integer c is performed during the exponentiation. She gathers many power consumptions cP_i with $i \in 1, 2, 3, \ldots$, and detects the spike arisen from the correlation function based on the specific bit of cP_i. The DPA can break Algorithm 3, because the sequence of generated points is deterministic and the DPA is able to find correlations for a specific bit.

Coron pointed out that it is necessary to insert random numbers during the computation of dP to prevent DPA [3]. The randomization eliminates the correlation between the secret bit and the sequence of points. The main idea of these countermeasures is to randomize the base point before starting the scalar multiplication. If the base point is randomized, there is no correlation among the power consumptions of each scalar multiplication. The DPA cannot obtain the spike of the power consumption derived from the statistical tool. This countermeasure is based on randomization of Jacobian coordinates. To prevent DPA we transform $P = (x, y)$ in affine coordinate to $P = (r^2x, r^3y, r)$ in Jacobian coordinates for a random value $r \in K^*$. This randomization produces the randomization in each representation of the point and the randomization of power consumptions during scalar multiplication dP.

However, Goubin proposed a DPA on Coron's randomization [5]. He pointed out that the point $(0, y)$ can not be randomized by Coron's randomization. Akishita and Takagi extended his attack to the case of auxiliary registers, called zero-value point attack [1]. The attack adaptively chooses a base point P and observes side channel information of the scalar multiplication dP, where d is a secret scalar. The bits of the secret scalar can be recovered if the point $(0, y)$ or zero-valued register appears. For example, the second most significant bit d_{n-1} should be 1 in Algorithm 3 if and only if for the point $(0, y)$ appears during the scalar multiplication dP with base point $P = (6^{-1}\#E)(0, y)$.

4.3 Proposed Attack

We propose an new attack on Algorithm 3 using the Coron's 3rd randomization.

Recall that the recommended curve from SECG uses the curve coefficient $a = -3$ [14]. If a is chosen as $a = -3$, the auxiliary parameter $M = 3X^2 + aZ^4$ of ECDBL in the Jacobian coordinate is computed by $M = 3(X + Z^2)(X - Z^2)$, and the computation time of ECDBL is reduced from $10M$ to $8M$, where M is the cost of a multiplication in K.

Assume that the underlying curve has the point P whose x-coordinate is equal to 2 (i.e., $(2, y)$). This point is randomized by the Coron's 3rd method: $(2r^2, r^3y, r)$ with a random element $r \in K$. Then the auxiliary parameter M takes value $3(2r^2 + r^2)(2r^2 - r^2) = 3(3r^2)(r^2)$. This means that ECDBL with input $(2, y)$ is not totally randomized by the Coron's 3rd method — there is an invariant of multiplication with the form $(3r^2)(r^2)$ under the Coron's 3rd randomization method.

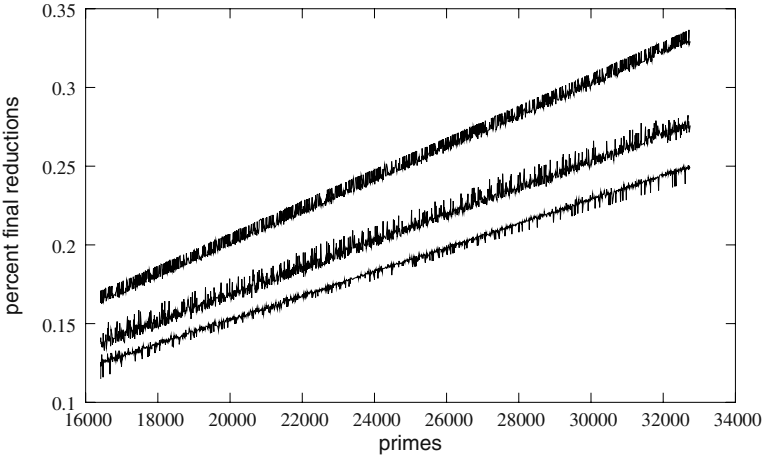

Fig. 1. The distribution of the final subtractions appeared in Montgomery multiplication for x^2 (upper), $x \cdot y$ with $y = 3 \cdot x \bmod m$ (middle), and $x \cdot y$ (lower)

Figure 1 shows the result of $m = 2^{15}$ and the primes run from 2^{14} up to m. It shows that we can statistically distinguish the operation $xy \bmod m$ with $y = 3x \bmod m$ from other operations (e.g., multiplication or squaring). The lowest curve shows the percentage of final subtractions which take place in the computation of $x \cdot y$ with $0 \le x, y < m$. The curve in the middle shows the results for $x \cdot y$ with $y = 3 \cdot x$ and the upper curve the results for x^2.

Because the distinction can be done a timing attack should be possible. The Coron's dummy method is vulnerable under the adaptive chosen cipher-text described in the previous section. We prove the distribution of the final subtraction appeared in $xy \bmod m$ with $y = 3x \bmod m$ in the following.

Theorem 2. *We assume that the assumption DIS is true. The final subtraction of Montgomery multiplication for xy with $y = 3x \bmod m$ asymptotically appears with probability $\frac{5m^2}{18R}$, where m is the underlying modulus.*

Proof. Let assume that $\gcd(3, m) = \gcd(3, R) = 1$ in this section. We consider the case of $y = 3x \bmod m$ in the following. Let $c(m, R)$ be the number of x that satisfies Equation (5):

$$c(m,R) := \#\{x \in \mathbb{Z} \mid 0 \le x \le m-1, y = 3x \bmod m,\ xy + (xym' \bmod R)m \ge mR\}.$$

We follow the estimation for the case of $x = y$. The number of integers $0 \le x \le m-1$ and whose images by $\phi_{m,3}(x) = x(3x \bmod m)$ are in the interval $[\xi R, (\xi+1)R)$ is

$$G_3(\xi) := \#\{x \in G_m \mid \phi_{m,3} \in [\xi R, (\xi+1)R)\}.$$

The function $\phi_{m,3}(x)$ is explicitly represented as follows:

$$\phi_{m,3}(x) = \begin{cases} 3x^2 & : \quad 0 \le x < \frac{m}{3} \\ x(3x-m) & : \quad \frac{m}{3} \le x < \frac{2m}{3} \\ x(3x-2m) & : \quad \frac{2m}{3} \le x < m. \end{cases}$$

Using the formula for solving quadratic equation, we can obtain the relationship:

$$G_3(\xi) \approx \begin{cases} \sqrt{\xi+1} - \sqrt{\xi} + \mu_1(\xi) - \mu_0(\xi) + \nu_1(\xi) - \nu_0(\xi) & : \quad 1 \le \xi < \frac{(m-1)^2}{3R} \\ \mu_1(\xi) - \mu_0(\xi) + \nu_1(\xi) - \nu_0(\xi) & : \quad \frac{(m-1)^2}{3R} \le \xi < \frac{2(m-1)^2}{3R} \\ \nu_1(\xi) - \nu_0(\xi) & : \quad \frac{2(m-1)^2}{3R} \le \xi < \frac{(m-1)^2}{3R}, \end{cases}$$

where $\mu_i(\xi) = \frac{\sqrt{m^2+12(\xi+i)R}}{6}$ and $\nu_i(\xi) = \frac{\sqrt{4m^2+12(\xi+i)R}}{6}$. From the same argument in the previous section, we are able to obtain the estimation about $c(m, R)$.

$$\begin{aligned} & c(m,R) \\ & \approx \frac{1}{m}\sqrt{\frac{R}{3}} \left(\sum_{\xi=1}^{\frac{(m-1)^2}{3R}} (\sqrt{x+1} - \sqrt{x}) + \sum_{\xi=1}^{\frac{2(m-1)^2}{3R}} (\mu_1(x) - \mu_0(x)) + \sum_{\xi=1}^{\frac{(m-1)^2}{R}} (\nu_1(x) - \nu_0(x)) \right) \end{aligned}$$

$$\approx \frac{1}{m}\sqrt{\frac{R}{3}}\left(\left(\frac{1}{3}\right)^{5/2}\left(\frac{m^2}{R}\right)^{3/2}+\frac{5\sqrt{3}}{54}\left(\frac{m^2}{R}\right)^{3/2}+\frac{4\sqrt{3}}{27}\left(\frac{m^2}{R}\right)^{3/2}\right)$$
$$\approx \frac{5}{18}\frac{m^2}{R}.$$

□

The average probability of occurring the final subtraction over randomly chosen K is $\frac{5}{24} = 0.208$, which is not equal to that of multiplication (0.188) or squaring (0.250). Similarly, we can prove that multiplication $x \cdot (ax)$ with small a has a different probability.

5 Conclusion

In this paper we presented some exact analysis related to the final subtraction of Montgomery multiplication. We investigated the distribution of outputs of multiplications in short intervals included between 0 and the underlying modulus. Integrating these results, we proved that the appearance ratios of the final subtraction during the Montgomery multiplication in the multiplication and squaring are asymptotically $\frac{m}{4R}$ and $\frac{m}{3R}$, respectively.

Based on the analysis we proposed a new invariant for the subtraction, namely multiplication $x \cdot (3x)$. We showed that this invariant appears at the randomization of parameter proposed by Coron, we could break it by DPA using the differences of the appearance ratios between general multiplications, squarings and the above case.

It is an interesting open problem to investigate further invariants of the Montgomery multiplication.

References

1. T. Akishita and T. Takagi, "Zero-Value Point Attacks on Elliptic Curve Cryptosystem", ISC 2003, LNCS 2851, pp.218-233, 2003.
2. H. Cohen, A. Miyaji, and T. Ono, "Efficient Elliptic Curve Exponentiation Using Mixed Coordinates", ASIACRYPT '98, LNCS 1514, pp. 51-65, 1998.
3. J.-S. Coron, "Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems", CHES '99, LNCS 1717, pp. 292-302, 2002.
4. J.-F. Dhem, F. Koeune, P.-A. Leroux, P. Mestré, J.-J. Quisquater, and J.-L. Willems, "A Practical Implementation of the Timing Attack," CARDIS 1998, LNCS 1820, pp.167-182, 2002.
5. L. Goubin, "A Refined Power-Analysis Attack on Elliptic Curve Cryptosystems", PKC 2003, LNCS 2567, pp. 199-211, 2003.
6. F. Koeune, W. Schindler and J-J. Quisqater "Improving Divide and Conquer Attacks against Cryptosystems by Better Error Detection / Correction Strategies," Cryptography and Coding 2001, Cirencester, LNCS 2260, 245-267, 2001
7. C. Kocher, "Timing attacks on Implementations of Diffie-Hellman, RSA, DSS, and other Systems," CRYPTO '96, LNCS 1109, pp.104-113, 1996.

8. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," CRYPTO '99, LNCS 1666, pp.388-397, 1999.
9. A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1997.
10. P. L. Montgomery, "Speeding the Pollard and Elliptic Curve Methods of Factorization", Mathematics of Computation, vol. 48, pp. 243-264, 1987.
11. W. Schindler, "A Timing Attack against RSA with the Chinese Remainder Theorem," CHES 2000, LNCS 1965, pp.109-124, 2000.
12. W. Schindler, "Optimized Timing Attacks against Public Key Cryptosystems," Statistic & Decisions 20, 191-210, 2002, R. Oldenbourg Verlag, Munich
13. W. Schindler and C. Walter, "More Detail for a Combined Timing and Power Attack against Implementations of RSA," IMA 2003, LNCS 2898, pp.245-263, 2003.
14. Standards for Efficient Cryptography Group (SECG), Specification of Standards for Efficient Cryptography. Available from `http://www.secg.org`
15. C. Walter, "Precise Bounds for Montgomery Modular Multiplication and Some Potentially Insecure RSA Moduli," CT-RSA 2002, LNCS 2271, pp.30-39, 2001.
16. C. Walter and S. Thompson, "Distinguishing Exponent Digits by Observing Modular Subtractions," CT-RSA 2001, LNCS 2020, pp.192-207, 2001.
17. G. Hachez and J.-J. Quisquater "Montgomery Exponentiation with no Final Subtraction: Improved Results," CHES 1999, LNCS 1965, pp.293-301, 1999.
18. C. D. Walter "Montgomery's Multiplication Technique: How to Make It Smaller and Faster," CHES 1999, LNCS 1965, pp.80-93, 1999.

Cryptography, Connections, Cocycles and Crystals: A p-Adic Exploration of the Discrete Logarithm Problem

H. Gopalkrishna Gadiyar, K. M. Sangeeta Maini, and R. Padma

AU-KBC Research Centre,
M.I.T. Campus of Anna University,
Chromepet, Chennai 600044, India
{gadiyar, maini, padma}@au-kbc.org

Abstract. Applying Hensel's lemma to the discrete logarithm problem over prime fields reveals the rich geometric and algebraic structure underlying the problem. It is shown that the problem has links to cocycles, connections, group extensions and crystalline cohomology. It is reminiscent of the recent use of Monsky-Washnitzer cohomology for counting points on hyperelliptic curves. Further some weak keys of the cryptosystems based on the hardness of the discrete logarithm problem over prime fields are discussed.

1 Introduction

The intractability of the discrete logarithm problem is the basis of the security of several cryptographic primitives. The Diffie - Hellman key exchange protocol is the first published public key cryptographic algorithm which is based on the difficulty of this problem. The El Gamal cryptosystem, the El Gamal signature scheme and the elliptic curve cryptosystem are a few other algorithms depending on the hardness of the discrete logarithm problem. The well known attacks on the problem like the baby step - giant step method, Pohlig - Hellman method and the index - calculus method are algebraic in nature. The first p-adic attacks on this hard problem were due to Smart [1], Semaev [2], Satoh and Araki [3] on anomalous elliptic curves. In [4] the p-adic methods are discussed from computer science perspective.

In this paper we apply the p-adic method to arrive at a very interesting structure underlying the discrete logarithm problem over prime fields. We find that the problem gets naturally linked to algebraic and geometric structures like the cocycles, connections, group extensions and crystalline cohomology. The body of the paper consists of four parts: In Section 2 a brief explanation of the p-adic attack on anomalous curves is given. In Section 3 we present the p-adic attack on non-anomalous curves. In addition we give a simplified version of the p-adic attack on the discrete logarithm problem over prime fields and explain why this attack fails. In Section 4 the failure of the attack is analyzed in detail and its connections to cocycles, connections, group extensions and crystalline

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 305–314, 2004.

cohomology are explained. It is found that the attack can be generalized to elliptic curve discrete logarithm problem (ECDLP). In Section 5 we use the results of Section 3 to demonstrate some weak keys.

2 The Anomalous Attack on ECDLP

Let E denote an elliptic curve over the p-adic field Q_p. The Weierstrass equation of E is given by

$$E \ : \ Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6 \, . \tag{1}$$

where $a_i \ \in Q_p$. Let $E(F_p)$ denote the elliptic curve after reduction of the coefficients of E modulo p. Let E_{ns} denote the set of nonsingular points of $E(F_p)$. Let

$$E_0(Q_p) = \{P \in E(Q_p) : \tilde{P} \in E_{ns}(F_p)\} \, , \tag{2}$$

and

$$E_1(Q_p) = \{P \in E(Q_p) : \tilde{P} = \tilde{O}\} \, , \tag{3}$$

where $\tilde{P}$ denotes the reduction point of P modulo p. One can also describe this group as follows.

$$E_1(Q_p) = \{P \in E(Q_p) : \mathrm{ord}_p(x(P)) \leq -2\} \, . \tag{4}$$

Also let

$$E_2(Q_p) = \{P \in E(Q_p) : \mathrm{ord}_p(x(P)) \leq -4\} \, , \tag{5}$$

where $x(P)$ denotes the x-coordinate of P. Let Z_p denote the ring of p-adic integers. The group $E_1(Q_p)$ is isomorphic to the group $\hat{E}(pZ_p)$ of pZ_p valued points of the one-parameter formal group associated to E and the isomorphism is given below.

$$\hat{E}(pZ_p) \rightarrow \qquad\qquad E_1(Q_p) \tag{6}$$

$$z \rightarrow \begin{cases} O \, , & \text{if } z \ = \ 0 \, , \\ (\frac{z}{w(z)}, \ -\frac{1}{w(z)}) \, , & \text{otherwise, i.e., } z \ = -\frac{x}{y} \, , \end{cases} \tag{7}$$

where $\omega(z)$ is a power series in z, which is the formal power series solution to the equation

$$\omega = z^3 + a_1z\omega + a_2z^2\omega + a_3\omega^2 + a_4z\omega^2 + a_6z\omega^3. \tag{8}$$

Using the power series for $\omega(z)$ the Laurent series for $x(z)$, $y(z)$, $w(z)$ can be computed, where $w(z)$ denotes the invariant differential on $\hat{E}(pZ_p)$. These Laurent series have their first few terms given by

$$x(z) = \frac{z}{\omega(z)} \ = \ \frac{1}{z^2} - \frac{a_1}{z} - a_2 - a_3z - (a_4 + a_1a_3)z^3 - \cdots , \tag{9}$$

$$y(z) = \frac{-1}{\omega(z)} = \frac{-1}{z^3} + \frac{a_1}{z^2} + \frac{a_2}{z} + a_3 + (a_4 + a_1a_3)z + \cdots , \tag{10}$$

$$w(z) = \frac{dx(z)}{2y(z) + a_1 x(z) + a_3} \tag{11}$$

$$= (1 + a_1 z + (a_1^2 + a_2)z^2 + (a_1^3 + 2a_1 a_2 + a_3)z^3 + \cdots)dz \tag{12}$$

$$= (1 + d_1 z + d_2 z^2 + d_3 z^3 + \cdots)dz\,. \tag{13}$$

For points on $E_1(Q_p)$ the p-adic elliptic logarithm is defined to be the group homomorphism

$$\vartheta_p : \begin{cases} E_1(Q_p) \to \hat{G}_a \\ P \qquad\quad \mapsto \int \omega(z_p) \;=\; z_p + \frac{d_1 z_p^2}{2} + \frac{d_2 z_p^3}{3} + \cdots\,. \end{cases} \tag{14}$$

The elliptic curve E is said to be an anomalous curve if $\#E(F_p) = p$. Smart, Semaev, Satoh and Araki gave a polynomial time algorithm to solve the discrete logarithm problem on such curves. We sketch the anomalous attack below.

Let $\tilde{P}$ and $\tilde{Q}$ be points on $E(F_p)$ such that $\tilde{Q} = [n]\tilde{P}$. The ECDLP is to find n given $\tilde{P}$ and $\tilde{Q}$. To do this, the p-adic approach was used. Let P and Q denote arbitrary lifts of the points $\tilde{P}$ and $\tilde{Q}$ on an elliptic curve over Q_p whose reduction mod p is the elliptic curve $E(F_p)$ where the lifting is done using Hensel's lemma [5]. In other words, P and $Q \in E_0(Q_p)$ and

$$Q - [n]P \in E_1(Q_p)\,. \tag{15}$$

We know that

$$E_0(Q_p)/E_1(Q_p) \cong E(F_p)\,, \tag{16}$$

and

$$E_1(Q_p)/E_2(Q_p) \cong F_p^+ \;(=\; (Z/pZ, +))\,. \tag{17}$$

What makes the anomalous attack work is due to the reason that $\#E(F_p) = p$. That is, we have the isomorphism

$$E_0(Q_p)/E_1(Q_p) \cong E_1(Q_p)/E_2(Q_p) \cong F_p^+\,. \tag{18}$$

Thus from (18) and (15), the points $[p]P$ and $[p]Q \in E_1(Q_p)$ (so that their elliptic logarithms exist) while their difference

$$[p]Q - [n][p]P \in E_2(Q_p)\,. \tag{19}$$

By taking elliptic logarithm ϑ_p on every term in (19) one gets

$$\vartheta_p([p]Q) - n\,\vartheta_p([p]P) \equiv 0 \bmod p^2\,. \tag{20}$$

Since $[p]P$, $[p]Q \in E_1(Q_p)$, and $\vartheta_p([p]P)$, $\vartheta_p([p]Q) \in pZ_p$, one obtains a linear congruence in n and the value of n can be computed. That is,

$$n \equiv \frac{\vartheta_p([p]Q)}{\vartheta_p([p]P)} \bmod p\,. \tag{21}$$

The elliptic logarithm is computed via the formal group variable and this can be done very easily. Hence one obtains n in polynomial time if $\vartheta_p([p]P) \neq 0$. In case $\vartheta_p([p]P) \equiv 0$, a different curve $E(Q_p)$ is chosen which reduces to $E(F_p)$ and the method is repeated.

3 The p-Adic Attack on Non-anomalous Curves

In this section we try to apply the anomalous attack on non-anomalous curves. There are two subsections where in Section 3.1 we give a general attack on non-anomalous curves followed by a discussion on the discrete logarithm problem over prime fields. In Section 3.2 we give a simplified p-adic attack on the same and the reason for its failure.

3.1 The p-Adic Attack on the Discrete Logarithm Problem Over Prime Fields

Let $\#E(F_p) = l\ (\neq p)$. From (16) we have $[l]P$ and $[l]Q \in E_1(Q_p)$ so that their elliptic logarithms exist. So we have

$$[l](Q - [n][P]) \in E_1(Q_p)\,, \tag{22}$$

but in general

$$[l](Q - [n][P]) \notin E_2(Q_p)\,, \tag{23}$$

while we still have (19) but $[p]Q$ and $[p]P \notin E_1(Q_p)$. Hence for the points $\tilde{P}$ and $\tilde{Q}$ related by $\tilde{Q} = [n]\tilde{P}$ if

$$[l](Q - [n][P]) \in E_2(Q_p)\,, \tag{24}$$

and if $[l]P$ and $[l]Q$ are not O, then the value of n can be found out immediately.

Let us apply the above discussion to the discrete logarithm problem over prime fields. Let a_0 be a primitive root of the prime p. The discrete logarithm problem in F_p is to find n given a_0 and b_0 where

$$a_0^n \equiv b_0 \bmod p\,. \tag{25}$$

It is well known that the multiplicative group F_p^* is isomorphic to the nonsingular points on the curve $y^2 + xy \equiv x^3 \bmod p$ and its cardinality is equal to $p-1$. In this case the formal group law is given by $F(z_1, z_2) = z_1 + z_2 - z_1 z_2$. Let us denote $2 \circ z = F(z, z)$ and $n \circ z = F((n-1) \circ z, z)$. Then $n \circ z = 1 - (1-z)^n$ and the map given by (7) is

$$z \to \begin{cases} O\,, & \text{if } z = 0\,, \\ \left(\frac{1-z}{z^2},\ -\frac{1-z}{z^3}\right), & \text{otherwise}\,. \end{cases} \tag{26}$$

If $x \in F_p^*$, then the corresponding formal group element $z = (1 - \frac{1}{x}) \bmod p$. If z_a and z_b correspond to arbitrary lifts of a_0 and b_0, then the left hand side of (22), written in terms of the formal group elements, is given by

$$(p-1) \circ z_b \; - \; n \circ (p-1) \circ z_a \,, \tag{27}$$

which is equal to

$$(1 - z_a)^{n(p-1)} \; - \; (1 - z_b)^{p-1} \,. \tag{28}$$

At this point we would like to stop and give a simplified approach to the discrete logarithm problem. One can easily see that the information we would get from (28) modulo p^2 is the same as what we would get by the method given below.

3.2 Simplified p-Adic Attack on the Discrete Logarithm Problem Over Prime Fields

Let $a_0 \in F_p^*$. We use the following notation for $a_0^n \bmod p^2$.

$$(a_0^n)_{p^2} \; = \; (a_0^n)_p \; + \; \beta_n p \,, \tag{29}$$

where β_n is the carry and it takes value between 0 and $p-1$ and the subscript p^k denotes the reduction modulo p^k. Hensel lifting a_0 and b_0 using the polynomial $x^p - x$ one gets $a_0 + a_1 p + a_2 p^2 + \cdots a_{k-1} p^{k-1}$ and $b_0 + b_1 p + b_2 p^2 + \cdots b_{k-1} p^{k-1}$ mod p^k, for any $k \geq 2$, where

$$a_{k-1} \equiv \frac{(a_0 + a_1 p + \cdots a_{k-2} p^{k-2})^p - (a_0 + a_1 p + \cdots a_{k-2} p^{k-2})}{p^{k-1}} \bmod p \,. \tag{30}$$

Also it satisfies the relation

$$(a_0 + a_1 p + \cdots a_{k-1} p^{k-1})^n \equiv (b_0 + b_1 p + \cdots b_{k-1} p^{k-1}) \bmod p^k \,. \tag{31}$$

In particular for $k = 2$, we have

$$a_1 \equiv \frac{a_0^p - a_0}{p} \bmod p \tag{32}$$

and

$$(a_0 + a_1 p)^n \; \equiv (b_0 + b_1 p) \bmod p^2 \,. \tag{33}$$

Expanding the left hand side of (33) and using (29), we get

$$(a_0^n)_p + \beta_n \; p + n \; (a_0)_p^{n-1} a_1 p \equiv b_0 + b_1 p \bmod p^2 \,. \tag{34}$$

But

$$(a_0^n)_p \; \equiv \; b_0 \bmod p \,. \tag{35}$$

Simplifying (34) one gets

$$n \frac{a_1}{a_0} \equiv \; \frac{b_1}{b_0} - \frac{\beta_n}{b_0} \bmod p \,. \tag{36}$$

Solving for n in terms of β_n gives

$$n \equiv \left(\frac{b_1 - \beta_n}{b_0}\right) / \left(\frac{a_1}{a_0}\right) \bmod p\,. \tag{37}$$

One can get this equation also by expanding $a_0^{n(p-1)} \bmod p^2$ in the following two ways: $((a_0)^n \bmod p^2)^{p-1} \bmod p^2$ and $((a_0)^{p-1} \bmod p^2)^n \bmod p^2$. This equation is a linear congruence in two unknowns, namely, n and β_n. So why the p-adic attack fails in this case is clear. The culprit is the carry term β_n. In anomalous case, $\beta_n = 0$. One can go back to (28) and expand the expression in two ways to see that one again gets (37).

4 Analysis of the p-Adic Attack

In this section we take the analysis of the last section in two directions:

1. Study the structure of β_n in (29).
2. Hensel lift (35) to infinite number of terms.

It will turn out that both approaches force us to look at an attack based on group extensions. This, according to us, is the essence of the attack pioneered by Smart, Semaev, Satoh and Araki.

4.1 Structure of β_n

We show here that the study of β_n leads us naturally to connections and group extensions.

Recall that ([6],[7]) if A is a ring, then a map $\delta: A \to A$ will be called a p-derivation if it satisfies

$$\delta(x+y) = \delta x + \delta y + C_p(x,y)\,, \tag{38}$$

and

$$\delta(xy) = x^p \delta y + y^p \delta x + p\delta x \delta y\,, \tag{39}$$

where $C_p(x,y) = \frac{(x^p+y^p-(x+y)^p)}{p}$ and $\delta x = \frac{(x-x^p)}{p}$. Note that in our notation a_1 is $-\delta(a_0) \bmod p$. Rewriting (36) in this notation, we get

$$\delta(a_0^n) \equiv \delta(b_0) + \beta_n \bmod p\,. \tag{40}$$

Note that in the parlance of differential geometry, β_n is nothing but a connection [8].

With the notation given in (29) we see that

$$(a_0^{n+m})_{p^2} \equiv ((a_0^n)_p + \beta_n p)((a_0^m)_p + \beta_m p) \bmod p^2 \tag{41}$$

$$\equiv (a_0^n)_p (a_0^m)_p + ((a_0^n)_p \beta_m + (a_0^m)_p \beta_n)p \bmod p^2\,, \tag{42}$$

while the left hand side is $(a_0^{n+m})_p + \beta_{n+m} p$. Hence

$$\beta_{n+m} \equiv (a_0^n)_p \beta_m + (a_0^m)_p \beta_n + \frac{1}{p}\{(a_0^n)_p (a_0^m)_p - (a_0^{n+m})_p\} \bmod p\,. \tag{43}$$

Let us define

$$\Delta(a_0^m, a_0^n) \equiv \frac{1}{p}\{\frac{(a_0^n)_p (a_0^m)_p}{(a_0^{n+m})_p} - 1\} \bmod p\,. \tag{44}$$

Then

$$\Delta(a_0^m, a_0^n) \equiv \frac{\beta_{n+m}}{(a_0^{n+m})_p} - \frac{\beta_n}{(a_0^n)_p} - \frac{\beta_m}{(a_0^m)_p} \bmod p\,. \tag{45}$$

One can immediately see that Δ is a 2-cocycle and hence is a trivial factor system. Recall [9] that if A and B are two commutative groups then any map $f : A \times A \to B$ satisfying the identity

$$f(y,z) - f(x+y,z) + f(x,y+z) - f(x,y) = 0,\ x,y,z \in A\,, \tag{46}$$

is called a factor system on A with values in B. If $g : A \to B$ is any map, the function δg defined by the formula

$$\delta g(x,y) = g(x+y) - g(x) - g(y)\,, \tag{47}$$

is a factor system; such a system is called trivial. The group of classes of factor systems modulo the trivial factor systems is denoted by $H^2(A,B)$. This leads us to the natural question: Can the discrete logarithm problem be related to group extensions? Recall [10] that a group extension (of A by B) is a short exact sequence

$$0 \to B \to E \to A \to 1\,, \tag{48}$$

of groups in which B is an abelian group. It is well known that $H^2(A,B)$ is isomorphic to the group of classes of central extensions of A by B. In our case, $A = (Z/pZ)^*$ and $B \cong (Z/pZ, +)$. So we find that the correct way is to view the p-adic attack as a problem in group extension.

For completeness of the discussion we add a further comment and refer the reader to the literature for details. A concrete realization of the extension problem is found in the standard construction of Witt vectors ([11], [12]) of length 2. There when one lifts the action of Frobenius one gets a function ψ which has all the desired properties:

$$\psi(x_1 + x_2) = \psi(x_1) + \psi(x_2) - \sum_{i=1}^{p-1} \frac{1}{p} \binom{p}{i} \overline{x}_1^i\, \overline{x}_2^{p-i}\,, \tag{49}$$

and

$$\psi(x_1 x_2) = \overline{x}_1^p\, \psi(x_2) + \overline{x}_2^p\, \psi(x_1) + p\, \psi(x_1)\, \psi(x_2)\,. \tag{50}$$

See [13] and the Appendix in [14]. In the next section we will find that we are led to crystalline cohomology by a totally different analysis. It is known that one of the ways of looking at crystalline cohomology is to view it as a group extension problem [15].

4.2 Hensel Lift, Teichmüller Character and Crystalline Cohomology

In Hensel lifting a_0 and b_0 as was done in Section 3.2, what we do actually is getting the Teichmüller representatives mod p^k [5] which is nothing but $a_0^{p^{k-1}} \bmod p^k$. Let $T(a_0)$ and $T(b_0)$ denote the Teichmüller representatives of a_0 and b_0 respectively in Z_p. If $a_0^n \equiv b_0 \bmod p$, then

$$T(a_0)^n = T(b_0)\,, \tag{51}$$

in Z_p. Also,

$$T(a_0)^{p-1} = 1 \text{ and } T(b_0)^{p-1} = 1\,, \tag{52}$$

in Z_p. We know that the logarithm is defined for a p-adic integer x if $\mathrm{ord}_p(x) \geq 1$ or in other words if $x \in pZ_p$. Iwasawa [5] defined the logarithm for any non-zero p-adic integer x as $\frac{1}{p-1} \log(x^{p-1})$. Note that $x^{p-1} \in 1 + pZ_p$ by Fermat's little theorem. Applying Iwasawa logarithm to (51), we get

$$n \log T(a_0) = \log T(b_0)\,. \tag{53}$$

Unfortunately both $\log T(a_0)$ and $\log T(b_0)$ are zero due to (52). So we get no information about n.

Teichmüller representatives are the $(p-1)^{th}$ roots of unity in Z_p^* and they lie in the kernel of the Iwasawa logarithm. It is clear that the Iwasawa logarithm corresponds to the real part of the p-adic logarithm. Hence we need a p-adic analogue of the complex logarithm which is provided by crystalline cohomology. One immediately notes that crystalline cohomology again deals with the extension of groups. See the essay by Günter Harder in [12] for a historical account and references to the original literature. More recent surveys on Internet are [16], [17].

So we are forced to one possible general attack based on p-adic methods which is an attack relying on the extension of groups. Such an attack could be extendable to elliptic curves also.

5 Some Implementation Issues

From the theoretical results we have got so far, we arrive at some weak keys of the cryptosystems based on the discrete logarithm problem over prime fields. By a weak key we mean that we can solve the corresponding discrete logarithm problem in polynomial time.

If the carry β_n in (37) is small, n can be found by trial and error. Few examples using ridiculously small primes are given in Table 1. The correct value of n is the one for which the columns b_0 and $a_0^n \ mod\ p$ are equal. We denote the correct value of n by $\surd$ and the wrong value by $\times$. Note that $\beta_n = 0$ if $a_0^n < p$. These are trivial weak keys while the examples we have provided are nontrivial. Also note that a_1 and b_1 can be computed in polynomial time.

Table 1

p	a_0	a_1	b_0	b_1	β	n	$a_0^n \ mod\ p$
11	2	10	7	3	0	7 $\surd$	7
103	5	85	63	67	0	77 $\times$	43
103	5	85	63	67	1	82 $\surd$	63
103	5	85	58	4	0	28 $\surd$	58

Thus from the above table one immediately observes that (a_0, b_0) with small β_n is a weak key. Here the calculations were done modulo p^2. One can naturally extend this to higher powers of p which we discuss next.

Let

$$(a_0 + a_1 p)^n \equiv b_0 + b_1 p + \gamma_n\ p^2 \bmod p^3\,, \tag{54}$$

and

$$(a_0 + a_1 p + a_2 p^2)^n \equiv b_0 + b_1 p + b_2 p^2 + \eta_n\ p^3 \bmod p^4\,. \tag{55}$$

Using (54)and (55) in (31) for $k = 3$ and 4 respectively and simplifying, we get

$$\frac{a_2}{a_0} n \ \equiv\ \frac{b_2}{b_0} - \frac{\gamma_n}{b_0} \bmod p\,, \tag{56}$$

and

$$\frac{a_3}{a_0} n \ \equiv\ \frac{b_3}{b_0} - \frac{\eta_n}{b_0} \bmod p\,. \tag{57}$$

If γ_n (η_n) is small, then n can be found by trial and error. We furnish some examples in Table 2 where we have only considered the cases $\gamma_n = 0$ and $\eta_n = 0$. One can go to higher powers of p similarly.

Table 2

p	a_0	a_1	a_2	a_3	b_0	b_1	b_2	b_3	n	$a_0^n \ mod\ p$
19	2	6	14	-	3	16	7	-	13	3
23	5	1	2	-	9	7	13	-	10	9
29	8	24	17	22	9	9	15	8	22	9
31	3	20	14	30	5	14	24	8	20	5
31	3	20	14	30	13	23	23	4	11	13

6 Conclusion

"Homological algebra is a tool used to prove nonconstructive existence theorems in algebra (and in algebraic topology). It also provides obstructions to carrying out various kinds of constructions; when the obstructions are zero, the construction is possible. Finally, it is detailed enough so that actual calculations may be performed in important cases." [10] It was a pleasant surprise to the authors that a simple tool like the Hensel's lemma led to such a deep structure. Thus the

discrete logarithm problem is brought into the fold of mainstream mathematical methods. The point counting algorithms using Monsky-Washnitzer cohomology [18] also seem to indicate that abstract mathematical tools may become common place in cryptanalysis in the near future.

References

1. N. P. Smart: The discrete logarithm problem on elliptic curves of trace one. J. Crypto. **12** (1999)193–196
2. I. A. Semaev: Evaluation of discrete logarithms on some elliptic curves. Math. Comp. **67** (1998) 353–356
3. T. Satoh and K. Araki: Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves. Comm. Math. Univ. Sancti Pauli **47** (1998) 81–92
4. D. Catalano, P. Q. Nguyen and J. Stern: The hardness of Hensel lifting: The case of RSA and discrete logarithm. ASIACRYPT 2002, Lecture Notes in Computer Science, Vol. 2501 (2002) 299–310
5. Alain M. Robert: A Course in p-adic Analysis. Springer - Verlag (2000) New York
6. A. Buium: Differential characters of abelian varieties over p-adic fields. Invent. Math. **122** (1995) 309–340
7. A. Buium: Arithmetic analogues of derivations. J. Algebra **198** (1997) 290-299
8. B. A. Dubrovin, A. T. Fomenko and S. P. Novikov: Modern Geometry - Methods and Applications. Part I. The Geometry of the surfaces, Transformation Groups, and Fields. Springer - Verlag (1984) New York
9. J. P. Serre: Algebraic Groups and Class Fields. Springer - Verlag (1988) New York
10. Charles A. Weibel: An Introduction to Homological Algebra. (1994) Cambridge University Press
11. J. P. Serre: Local Fields. Springer - Verlag (1979) New York
12. Ernst Witt: Collected Papers. Springer - Verlag (1998) Berlin - Heidelberg
13. P. Deligne and L.Illusie: Relèvements modulo p^2 et décomposition du complexe de de Rham. Invent. Math. **89** (1987) 247–270
14. V. B. Mehta and V. Srinivas: Varieties in positive characteristic with trivial tangent bundle. Compos. Math. **64**, (1987) 191–212. (Appendix on Canonical Liftings by M. V. Nori and V. Srinivas.)
15. B. Mazur and W. Messing: Universal Extensions and One Dimensional Crystalline Cohomology. Lecture Notes in Mathematics, **370** (1974) Springer
16. Yves André: Period mappings and differential equations. From C to C_p. Tôhoku - Hokkaidô lectures in Arithmetic Geometry. (With appendices by F. Kato and N. Tsuzuki.) arXiv:math.NT/0203194v1 19 March 2002
17. Marc - Hubert Nicole: Cris is for Crystalline. (Reference unavailable. Downloaded from Internet.)
18. Kiran S. Kedlaya: Counting points on hyperelliptic curves using Monsky-Washnitzer cohomology. J. Ramanujan Math. Soc. **16** (2001) 323-338

EME*: Extending EME to Handle Arbitrary-Length Messages with Associated Data

Shai Halevi

IBM T.J. Watson Research Center, Hawthorne, NY 10532, USA
shaih@alum.mit.edu

Abstract. This work describes a mode of operation, EME*, that turns a regular block cipher into a length-preserving enciphering scheme for messages of (almost) arbitrary length. Specifically, the resulting scheme can handle any bit-length, not shorter than the block size of the underlying cipher, and it also handles associated data of arbitrary bit-length. Such a scheme can either be used directly in applications that need encryption but cannot afford length expansion, or serve as a convenient building block for higher-level modes.

The mode EME* is a refinement of the EME mode of Halevi and Rogaway, and it inherits the efficiency and parallelism from the original EME.

1 Introductions

Adding secrecy protection to existing (legacy) protocols and applications raises some unique problems. One of these problems is that existing protocols sometimes require that the encryption be "transparent", and in particular preclude length-expansion. One example is encryption of storage data "at the sector level", where both the higher-level operating system and the lower-level disk expect the data to be stored in blocks of 512 bytes, and so any encryption method would have to accept 512-byte plaintext and produce 512-byte ciphertext.

Clearly, insisting on a length-preserving (and hence deterministic) transformation has many drawbacks. Indeed, even the weakest common notion of security for "general purpose encryption" (i.e., semantic security [6]) cannot be achieved by deterministic encryption. Still, there may be cases where length-preservation is a hard requirement (due to technical, economical or even political constrains), and in such cases one may want to use some encryption scheme that gives better protection than no encryption at all. The strongest notions of security for a length-preserving transformation is "strong pseudo-random permutation" (SPRP) as defined by Luby and Rackoff [12], and its extension to "tweakable SPRP" by Liskov et al. [11]. A "tweak" is an additional input to the enciphering and deciphering procedures that need not be kept secret. This report uses the terms "tweak" and "associated data" pretty much interchangeably, except that "associated data" hints that it can be of arbitrary length, whereas "tweak" is sometimes thought of as a fixed-length quantity.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 315–327, 2004.

Motivated by the application for "sector level encryption", some efficient modes of operation that implement "tweakable SPRP" on large blocks were recently described by Halevi and Rogaway [7, 8]. As "general purpose modes", however, these modes are somewhat limited, in that they can only be applied to input messages whose size is a multiple of n, the block-size of the underlying cipher. Also, the mode CMC from [7] is inherently sequential (and it was only proven secure against attack model where all the messages are of the same length), and the mode EME from [8] is limited to messages of at most n^2 bits. The current work is aimed at eliminating these limitations.

The mode EME*, presented below, takes a standard cipher with n-bit blocks and turns it into a tweakable enciphering scheme with message space $\mathcal{M} = \{0,1\}^{n+}$ (i.e., any string of at least n bits) and tweak space $\mathcal{T} = \{0,1\}^*$. The key for EME* consists of one key of the underlying cipher and two additional n-bit blocks. The mode EME* has similar structure to the mode EME from [8]. Roughly, it consists of two layers of masked ECB encryption, with a layer of "lightweight mixing" in between. As a consequence, EME* is highly parallelizeable,[1] and also quite work-efficient. Processing an m-block query with ℓ blocks of associated data takes at most $\ell + 2m + \lceil m/n \rceil$ block encryptions (or decryptions).

1.1 Design Alternatives

There are a few different practical approaches for transforming a standard n-bit block cipher into a variable-input-length cipher. As described above, the approach that we use in this work is the Encrypt-mix-Decrypt approach that was used in CMC and EME.

A different approach is based on the Luby-Rackoff construction [12] (aka Feistel network). This approach was examined in the work of Petal et al. [17], and was also adopted by the ABL mode of McGrew and Viega [13]. The Luby-Rackoff construction involves splitting the m-bit input into two parts (say, of lengths m_l and m_r bits), and using the underlying cipher to implement a two pseudorandom functions: one from m_l to m_r bits and the other from m_r to m_l bits. The pseudorandom functions can be built, for example, by first applying CBC-MAC to the input to get an n-bit block, and then expand this block to the output size using counter mode. This approach (in its simplest form) takes four applications of these pseudorandom functions (two for each function), and each of these applications takes time roughly equivalent to an encryption of the entire input. Such straightforward implementation would therefore take roughly twice the work of EME*. One could instead use the modification of Naor and Reingold [15], where the first and last application are replaces by universal hashing. However, as was discussed in the CMC paper [7], universal hashing with sufficiently

[1] In EME*, the longest execution path for any input consists of at most five block encryption. If the input length is a multiple of the block length then the longest path has only four encryptions. If in addition the input is shorter than n blocks then the longest path has only three encryptions.

small collision probability is quite hard to implement, and it is not clear to what extent this improves the performance on common real-world platforms[2].

A somewhat better solution would be to use a construction due to Naor and Reingold [14], that involves one layer of encryption "sandwiched" between two layers of universal hashing. As with the previous approach, here too one runs into the problem of implementing universal hashing with small enough collision probability. (And if the underlying block cipher is used to implement these hashing layers then one gets three layers of encryption, which is about 50% more work than EME*.)

1.2 What About Very Short Blocks?

The mode EME* can handle blocks of any bit-length *but not less that the block size of the underlying cipher*. The underlying structure of EME*, being based on ECB encryption, does not lend itself to handling shorter blocks. In fact, in my opinion there is no good solution today for handling arbitrary short blocks. The solutions that I am aware of are described below. (The first three of those were described in a paper of Black and Rogaway [3].)

- For blocks that are only slightly shorter than the block length of the underlying cipher (e.g., 127-bit blocks), one could use the following trick: First, pad the plaintext to one block and use directly the underlying block cipher. If the ciphertext does not fall in the right domain (e.g., you need a 127-bit block but the ciphertext does not begin with zero), then encrypt it again. Repeat as many times as needed to get the ciphertext in the right domain. It can be shown that this procedure results in a well defined permutation. The drawback is that when using this method to encrypt a block of size 2^{n-m}, the number of applications of the underlying block cipher is 2^m on the average. (Here n is the block length of the underlying cipher.)
- For very short blocks (e.g., one byte) it is possible to pre-compute a pseudorandom permutation and store it in a table. This approach, however, clearly runs out of steam for blocks longer than two bytes, and it is extremely wasteful of space even before that. (Also, it is not clear how to incorporate a "tweak" into this approach.)
- Alternatively, one could apply the Luby-Rackoff construction to implement the narrow-block cipher, using the underlying cipher for the pseudorandom functions. This solution extends to handle messages of any length, but at a price of a severely reduced security-parameter. For example, although 128-bit blocks may enjoy "128 bits of security", 127-bit blocks only enjoy "63 bits of security". Even worse, 64-bit blocks have to make due with a pathetic "32 bits of security".

  It is possible to use six or more rounds of the Luby-Rackoff construction to make the security parameter a little less miserable (cf. Patarin's work [16]), but the price is an extremely slow mode for small blocks.

[2] Another minor drawback of the Luby-Rackoff construction is that for inputs of length under two blocks, the security parameter is less that the one offered by either EME* or the Naor-Reingold construction.

– For blocks that are not too short (say, at least 64 bits), one can simply switch to using a different block cipher. For example, one could use EME*[AES] to process blocks that are 128 bits or more, and use a separately keyed EME*[3DES] to handle blocks of length between 64 and 127 bits.

  This solution, however, is quite expensive, as it mandates the implementation of two different ciphers. (Of course, one could use EME*[3DES] also to handle longer messages, but then the security parameter would be much reduced.) Moreover this solution does not address blocks shorter than 64 bits.
– Another approach is to use a parameterizable cipher (e.g., RC5 [18]) as the underlying block cipher. Parameterizable ciphers can be instantiated to handle various block sizes, so in particular they can be used in their narrow-block instantiation to handle the small blocks. However, to the best of my knowledge there is a fairly small number of such ciphers, and they were never seriously analyzed for small blocks. So it unlikely that they provide very good security, especially in the very small block sizes. Worse still, it is likely that using the same key for different block sizes would have disastrous consequences.

I view the problem of handling arbitrary small blocks as wide open. The two plausible approaches for addressing it are either to design a mode of operation with good security-performance tradeoff for small blocks, or to design an efficient block cipher that can handle small blocks securely. I believe that a good cipher is more likely to be possible than a good mode of operation (but perhaps this is only because I know more about modes of operation than about block ciphers.)

Organization. Section 2 recalls some standard definitions (this section is taken almost verbatim from [8]). Section 3 describes the EME* mode with a brief discussion of the extensions of EME* over EME. The security of EME* is stated in Section 4. The proof, which is long and technical, can be found in the full version of this report [9]. Some aspects of that proof are discussed in the appendix.

2 Preliminaries

A *tweakable enciphering scheme* is a function $\mathbf{E}$: $\mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\mathcal{M} = \bigcup_{i \in I} \{0,1\}^i$ is the *message space* (for some nonempty index set $I \subseteq \mathbb{N}$) and $\mathcal{K} \neq \emptyset$ is the *key space* and $\mathcal{T} \neq \emptyset$ is the *tweak space*. We require that for every $K \in \mathcal{K}$ and $T \in \mathcal{T}$ we have that $\mathbf{E}(K,T,\cdot) = \mathbf{E}_K^T(\cdot)$ is a length-preserving permutation on $\mathcal{M}$. The inverse of an enciphering scheme $\mathbf{E}$ is the enciphering scheme $\mathbf{D} = \mathbf{E}^{-1}$ where $X = \mathbf{D}_K^T(Y)$ if and only if $\mathbf{E}_K^T(X) = Y$. A *block cipher* is the special case of a tweakable enciphering scheme where the message space is $\mathcal{M} = \{0,1\}^n$ (for some $n \geq 1$) and the tweak space is $\mathcal{T} = \{\varepsilon\}$ (the empty string). The number n is called the *blocksize*. By $\mathrm{Perm}(n)$ we mean the set of all permutations on $\{0,1\}^n$. By $\mathrm{Perm}^{\mathcal{T}}(\mathcal{M})$ we mean the set of all functions π: $\mathcal{T} \times \mathcal{M} \to \mathcal{M}$ where $\pi(T,\cdot)$ is a length-preserving permutation.

An *adversary* A is a (possibly probabilistic) algorithm with access to some oracles. Oracles are written as superscripts. By convention, the running time of an algorithm includes its description size. The notation $A \Rightarrow 1$ describes the event that the adversary A outputs the bit one.

Security Measure. For a tweakable enciphering scheme $\mathbf{E}\colon \mathcal{K} \times \mathcal{T} \times \mathcal{M} \to \mathcal{M}$ we consider the advantage that the adversary A has in distinguishing $\mathbf{E}$ and its inverse from a random tweakable permutation and its inverse: $\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathbf{E}}(A) =$

$$\Pr\left[K \xleftarrow{\$} \mathcal{K}\colon\ A^{\mathbf{E}_K(\cdot,\cdot)\,\mathbf{E}_K^{-1}(\cdot,\cdot)} \Rightarrow 1\right] - \Pr\left[\pi \xleftarrow{\$} \mathrm{Perm}^{\mathcal{T}}(\mathcal{M})\colon\ A^{\pi(\cdot,\cdot)\,\pi^{-1}(\cdot,\cdot)} \Rightarrow 1\right]$$

The notation shows, in the brackets, an experiment to the left of the colon and an event to the right of the colon. We are looking at the probability of the indicated event after performing the specified experiment. By $X \xleftarrow{\$} \mathcal{X}$ we mean to choose X at random from the finite set $\mathcal{X}$. In writing $\pm\widetilde{\mathrm{prp}}$ the tilde serves as a reminder that the PRP is tweakable and the $\pm$ symbol is a reminder that this is the "strong" (chosen plaintext/ciphertext attack) notion of security. For a block cipher, we omit the tilde.

Without loss of generality we assume that an adversary never repeats an encipher query, never repeats a decipher query, never queries its deciphering oracle with (T, C) if it got C in response to some (T, M) encipher query, and never queries its enciphering oracle with (T, M) if it earlier got M in response to some (T, C) decipher query. We call such queries *pointless* because the adversary "knows" the answer that it should receive.

When $\mathcal{R}$ is a list of resources and $\mathbf{Adv}^{\mathrm{xxx}}_{\Pi}(A)$ has been defined, we write $\mathbf{Adv}^{\mathrm{xxx}}_{\Pi}(\mathcal{R})$ for the maximal value of $\mathbf{Adv}^{\mathrm{xxx}}_{\Pi}(A)$ over all adversaries A that use resources at most $\mathcal{R}$. Resources of interest are the running time t, the number of oracle queries q, and the query complexity parameters σ^a_n and σ^d_n (where $n \geq 1$ is a number). The parameters σ^a_n, σ^d_n are the total number of n-bit blocks in all the associated-data and data portions, respectively, in all queries that the adversary makes. Namely, the query complexity parameters of any one call (T, P) are $\sigma^a_n = \lceil |T|/n \rceil$ and $\sigma^d_n = \lceil |P|/n \rceil$, and the query complexity parameters of an attack are the respective sums of the query complexity parameters of all the calls. The name of an argument (e.g., t, q, σ^a_n, σ^d_n) will be enough to make clear what resource it refers to.

Finite Fields. We interchangeably view an n-bit string as: a string; a nonnegative integer less than 2^n (msb first); a formal polynomial over GF(2) (with the coefficient of x^{n-1} first and the free term last); and an abstract point in the finite field $\mathrm{GF}(2^n)$. To do addition on field points, one xors their string representations. To do multiplication on field points, one must fix a degree-n irreducible polynomial. We choose to use the lexicographically first primitive polynomial of minimum weight. For $n = 128$ this is the polynomial $\mathsf{x}^{128} + \mathsf{x}^7 + \mathsf{x}^2 + \mathsf{x} + 1$. See [4] for a list of the indicated polynomials. We note that with this choice of field-point representations, the point $\mathsf{x} = 0^{n-2}10 = 2$ will always have order $2^n - 1$ in the multiplicative group of $\mathrm{GF}(2^n)$, meaning that $2, 2^2, 2^3, \ldots, 2^{2^n-1}$ are all distinct. Finally,

we note that given $L = L_{n-1} \cdots L_1 L_0 \in \{0,1\}^n$ it is easy to compute $2L$. We illustrate the procedure for $n = 128$, in which case $2L = L{\ll}1$ if $\mathsf{firstbit}(L) = 0$, and $2L = (L{\ll}1) \oplus \mathrm{Const87}$ if $\mathsf{firstbit}(L) = 1$. Here $\mathrm{Const87} = 0^{120}10^41^3$ and $\mathsf{firstbit}(L)$ means L_{n-1} and $L{\ll}1$ means $L_{n-2}L_{n-3} \cdots L_1 L_0 0$.

3 Specification of EME* Mode

Consider a block cipher E: $\mathcal{K}{\times}\{0,1\}^n \to \{0,1\}^n$. Then EME*$[E]$: $(\mathcal{K}{\times}\{0,1\}^{2n}){\times}\mathcal{T} \times \mathcal{M} \to \mathcal{M}$ is an enciphering scheme with associated data, where $\mathcal{K}$ is the same as the underlying cipher, $\mathcal{T} = \{0,1\}^{0..n(2^n-3)}$, and $\mathcal{M} = \{0,1\}^{n..n(2^n-2)}$. In words, the key for EME*$[E]$ consists of one key K of the underlying block cipher E and two n-bit blocks, L and R. EME*$[E]$ accepts messages of any bit length grater than or equal to n (but no more than $n(2^n - 2)$), and associated data of arbitrary bit-length (but no more than $n(2^n - 3)$). Obviously, in practical terms the upper limits are no limitation at all.

The scheme EME*$[E]$ follows the same general principles of the tweakable scheme EME from [8]. Roughly, it consists of two layers of masked ECB encryption, with a layer of "lightweight mixing" in between. A complete specification of the enciphering scheme EME*$[E]$ is given in Figure 1, and an illustration (for a message of $n+2$ full blocks and one partial block) is provided in Figure 2. For those familiar with EME, the differences between EME and EME* are as follows:

- *Hashing the "tweak".* The original EME scheme requires that the "tweak value" be an n-bit string, whereas here we allow associated data of any length. For this purpose, we hash the associated data to an n-bit string. The hash function need only be xor-universal, yet it is implemented using the underlying block cipher in a PMAC-like mode [2].
- *More than one mask.* The EME scheme uses (multiples of) a single mask value M in the "lightweight masking" layer. It was shown in [8], however, that this masking technique with just one mask cannot be used for messages longer than n^2 bits.

  Longer messages are handled in EME* using the approach that was proposed in the appendix of [8]. The message is broken to chunks of at most n^2 bits each, and a different mask value is used for every chunk. To handle the last partial block (if any), yet another mask is computed and xor-ed into the last partial plaintext block, thus getting the last partial ciphertext block.

We comment that it is possible to derive the two key blocks L, R from the cipher key K, say by setting $L = 2E_K(0)$ and $R = 3E_K(0)$[3]. This variant is not proven in the current work, but I believe that such a proof is possible (although no means trivial).

[3] The maximum length of messages and associated input would have to be somewhat reduced for this to work. But for $n = 128$ we can still prove security for messages and associated data as long as, say, 2^{120} blocks. (The upper bound is actually $\min(\log_2 3,\ 2^n - 1 - \log_2 3)$. With the representation of $GF(2^{128})$ as above, we have $\log_2 3 \approx 3.39 \times 10^{38} \approx 2^{128} - 2^{120}$. See [19].)

function $H_{K,R}(T_1 \cdots T_{\ell-1}, T_\ell)$: // $|T_1| = \cdots = |T_{\ell-1}| = n$, $0 < |T_\ell| \le n$

01 **if** T is empty **return** $E_K(R)$

10 **for** $i \in [1..\ell-1]$ **do** $TTT_i \leftarrow E_K(2^i R \oplus T_i) \oplus 2^i R$
11 **if** $|T_\ell| = n$ **then** $TTT_\ell \leftarrow E_K(2^\ell R \oplus T_\ell) \oplus 2^\ell R$
12 **else** $TTT_\ell \leftarrow E_K(2^{\ell+1} R \oplus (T_\ell 10..0)) \oplus 2^{\ell+1} R$
13 **return** $TTT_1 \oplus \cdots \oplus TTT_\ell$

Algorithm $\mathbf{E}_{K,L,R}(T; P_1 \cdots P_m)$

// $|P_1| = \cdots = |P_{m-1}| = n$, $0 < |P_m| \le n$

101 **if** $|P_m| = n$ **then** *lastFull* $\leftarrow m$
102 **else** *lastFull* $\leftarrow m - 1$
103 $PPP_m \leftarrow P_m$ padded with 10..0

110 **for** $i \leftarrow 1$ **to** *lastFull* **do**
111 $PP_i \leftarrow 2^{i-1} L \oplus P_i$
112 $PPP_i \leftarrow E_K(PP_i)$

120 $SP \leftarrow PPP_2 \oplus \cdots \oplus PPP_m$
121 $MP_1 \leftarrow PPP_1 \oplus SP \oplus H_{K,R}(T)$
122 **if** $|P_m| = n$ **then** $MC_1 \leftarrow E_K(MP_1)$
123 **else** $MM \leftarrow E_K(MP_1)$
124 $MC_1 \leftarrow E_K(MM)$
125 $C_m \leftarrow P_m \oplus (MM$ truncated$)$
126 $CCC_m \leftarrow C_m$ padded with 10..0
127 $M_1 \leftarrow MP_1 \oplus MC_1$

130 **for** $i = 2$ **to** *lastFull* **do**
131 $j = \lceil i/n \rceil$, $k = (i-1) \bmod n$
132 **if** $k = 0$ **then**
133 $MP_j \leftarrow PPP_i \oplus M_1$
134 $MC_j \leftarrow E_K(MP_j)$
135 $M_j \leftarrow MP_j \oplus MC_j$
136 $CCC_i \leftarrow MC_j \oplus M_1$
137 **else** $CCC_i \leftarrow PPP_i \oplus 2^k M_j$

140 $SC \leftarrow CCC_2 \oplus \cdots \oplus CCC_m$
141 $CCC_1 \leftarrow MC_1 \oplus SC \oplus H_{K,R}(T)$
142 **for** $i \leftarrow 1$ **to** *lastFull* **do**
143 $CC_i \leftarrow E_K(CCC_i)$
144 $C_i \leftarrow CC_i \oplus 2^{i-1} L$

150 **return** $C_1 \ldots C_m$

Algorithm $\mathbf{D}_{K,L,R}(T; C_1 \cdots C_m)$

// $|C_1| = \cdots = |C_{m-1}| = n$, $0 < |C_m| \le n$

201 **if** $|C_m| = n$ **then** *lastFull* $\leftarrow m$
202 **else** *lastFull* $\leftarrow m - 1$
203 $CCC_m \leftarrow C_m$ padded with 10..0

210 **for** $i \leftarrow 1$ **to** *lastFull* **do**
211 $CC_i \leftarrow 2^{i-1} L \oplus C_i$
212 $CCC_i \leftarrow E_K^{-1}(CC_i)$

220 $SC \leftarrow CCC_2 \oplus \cdots \oplus CCC_m$
221 $MC_1 \leftarrow CCC_1 \oplus SC \oplus H_{K,R}(T)$
222 **if** $|C_m| = n$ **then** $MP_1 \leftarrow E_K^{-1}(MC_1)$
223 **else** $MM \leftarrow E_K^{-1}(MC_1)$
224 $MP_1 \leftarrow E_K^{-1}(MM)$
225 $P_m \leftarrow C_m \oplus (MM$ truncated$)$
226 $PPP_m \leftarrow P_m$ padded with 10..0
227 $M_1 \leftarrow MP_1 \oplus MC_1$

230 **for** $i = 2$ **to** *lastFull* **do**
231 $j = \lceil i/n \rceil$, $k = (i-1) \bmod n$
232 **if** $k = 0$ **then**
233 $MC_j \leftarrow CCC_i \oplus M_1$
234 $MP_j \leftarrow E_K^{-1}(MC_j)$
235 $M_j \leftarrow MP_j \oplus MC_j$
236 $PPP_i \leftarrow MP_j \oplus M_1$
237 **else** $PPP_i \leftarrow CCC_i \oplus 2^k M_j$

240 $SP \leftarrow PPP_2 \oplus \cdots \oplus PPP_m$
241 $PPP_1 \leftarrow MP_1 \oplus SP \oplus H_{K,R}(T)$
242 **for** $i \leftarrow 1$ **to** *lastFull* **do**
243 $PP_i \leftarrow E_K^{-1}(PPP_i)$
244 $P_i \leftarrow PP_i \oplus 2^{i-1} L$

250 **return** $P_1 \ldots P_m$

Fig. 1. Enciphering and deciphering under $\mathbf{E} = \text{EME}^*[E]$, where $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ is a block cipher. The associated data is $T \in \{0,1\}^*$, the plaintext is $P = P_1 \cdots P_m$ and the ciphertext is $C = C_1 \cdots C_m$

4 Security of EME*

The following theorem relates the advantage of an adversary in attacking $\text{EME}^*[E]$ to the advantage an adversary in attacking the block cipher E.

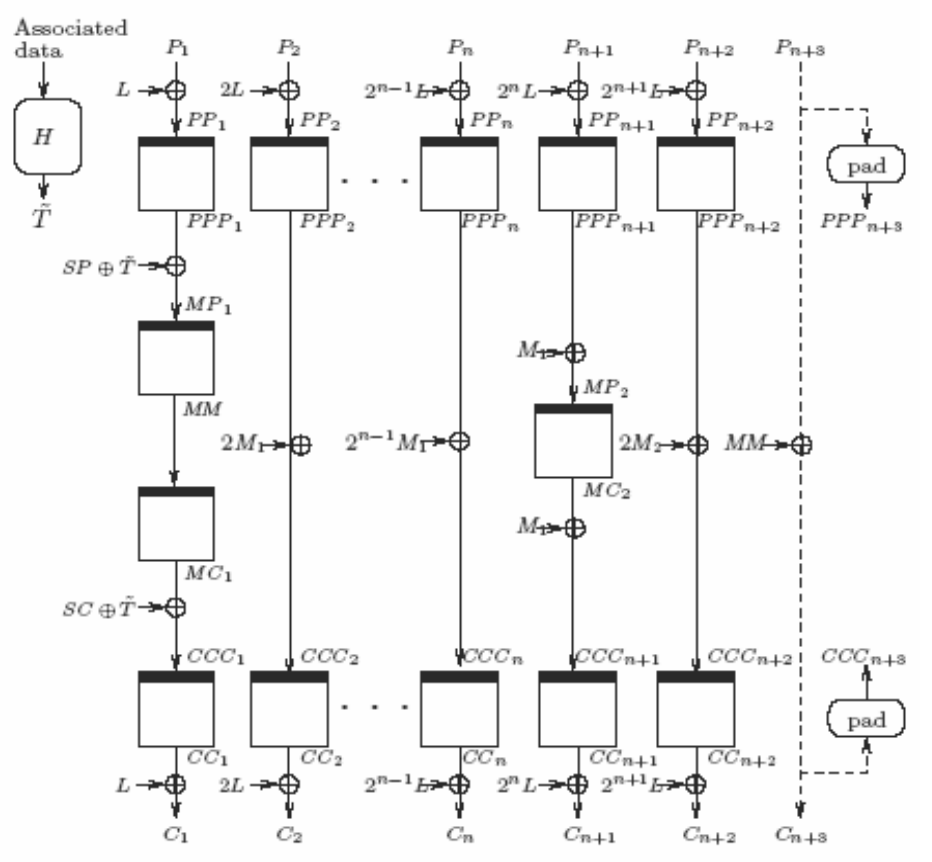

Fig. 2. Enciphering under EME* a buffer with $n + 2$ full blocks and one partial block. The boxes represent E_K. We set the masks as $SP = PPP_2 \oplus \cdots \oplus PPP_{n+3}$, $M_i = MP_i \oplus MC_i$, and $SC = CCC_2 \oplus \cdots \oplus CCC_{n+3}$

Theorem 1. [EME* security] *Fix $n \geq 4$ and $q, \sigma_n^a, \sigma_n^d \in \mathbb{N}$. Any adversary that attacks* $\mathrm{EME}^*[\mathrm{Perm}(n)]$*, trying to distinguish it from a truly random tweakable length-preserving permutation, using at most q queries totaling at most σ_n^a blocks of associated data and σ_n^d blocks of data, has advantage at most $(\sigma_n^a + 2.25(\sigma_n^d + q))^2/2^n$. Using the notations from Section 2, we have*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{EME}^*[\mathrm{Perm}(n)]}(q, \sigma_n^a, \sigma_n^d) \leq \frac{(\sigma_n^a + 2.25(\sigma_n^d + q))^2}{2^n} \tag{1}$$

Corollary 1. *Fix $n \geq 4$ and $t, q, \sigma_n^a, \sigma_n^d \in \mathbb{N}$ and a block cipher $E\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$. Then*

$$\mathbf{Adv}^{\pm\widetilde{\mathrm{prp}}}_{\mathrm{EME}^*[E]}(t, q, \sigma_n^a, \sigma_n^d) \le \frac{(\sigma_n^a + 2.25(\sigma_n^d + q))^2}{2^n} + 2\,\mathbf{Adv}^{\pm\mathrm{prp}}_{E}\left(t',\ \sigma_n^a + q + 2.25\sigma_n^d\right)$$

where $t' = t + O(n(\sigma_n^a + \sigma_n^d))$. □

Note that the theorem and corollary *do not* restrict messages to one particular length: proven security is for a variable-input-length (VIL) cipher, not just fixed-input-length (FIL) one. The proof of Theorem 1 is given in the full version of this paper [9], and some aspects of it are discussed in Appendix A. Corollary 1 embodies the standard way to pass from the information-theoretic setting to the complexity-theoretic one.

Acknowledgements. I thank John Viega for showing me his ABL mode of operation, and Eli Biham for a discussion about the state of block ciphers for very short blocks. I also thank the anonymous Indocrypt referees for their useful comments.

References

1. J. Black and P. Rogaway. CBC MACs for arbitrary-length messages: The three-key constructions. In *Advances in Cryptology – CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 197–215. Springer-Verlag, 2000.
2. J. Black and P. Rogaway. A block-cipher mode of operation for parallelizable message authentication. In L. Knudsen, editor, *Advances in Cryptology – EUROCRYPT '02*, volume 2332 of *Lecture Notes in Computer Science*, pages 384–397. Springer-Verlag, 2002.
3. J. Black and P. Rogaway. Ciphers with arbitrary finite domains. In *The RSA conference – Cryptographer's track, RSA-CT'02*, volume 2271 of *Lecture Notes in Computer Science*, pages 114–130. Springer-Verlag, 2002.
4. S. Duplichan. A primitive polynomial search program. Web document. Available at http://users2.ev1.net/∼sduplichan/primitivepolynomials/primivitePolynomials.htm, 2003.
5. S. Even and Y. Mansour. A construction of a cipher from a single pseudorandom permutation. *Journal of Cryptology*, 10(3):151–162, 1997.
6. S. Goldwasser and S. Micali. "Probabilistic encryption". *J. of Computer and System Sciences*, 28, April 1984.
7. S. Halevi and P. Rogaway. A tweakable enciphering mode. In D. Boneh, editor, *Advances in Cryptology – CRYPTO '03*, volume 2729 of *Lecture Notes in Computer Science*, pages 482–499. Springer-Verlag, 2003. Full version available on the ePrint archive, http://eprint.iacr.org/2003/148/.
8. S. Halevi and P. Rogaway. A parallelizable enciphering mode. In *The RSA conference – Cryptographer's track, RSA-CT'04*, volume 2964 of *Lecture Notes in Computer Science*, pages 292–304. Springer-Velrag, 2004. Full version available on the ePrint archive, http://eprint.iacr.org/2003/147/.
9. S. Halevi. EME*: extending EME to handle arbitrary-length messages with associated data. In *Indocrypt 2004*. Full version available on the ePrint archive, http://eprint.iacr.org/2004/125/.

10. J. Kilian and P. Rogaway. How to protect DES against exhaustive key search. *Journal of Cryptology*, 14(1):17–35, 2001. Earlier version in CRYPTO '96. www.cs.ucdavis.edu/∼rogaway.
11. M. Liskov, R. Rivest, and D. Wagner. Tweakable block ciphers. In *Advances in Cryptology – CRYPTO '02*, volume 2442 of *Lecture Notes in Computer Science*, pages 31–46. Springer-Verlag, 2002. www.cs.berkeley.edu/∼daw/.
12. M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2), April 1988.
13. D. A. McGrew and J. Viega. ABL mode: security without data expansion. Private communication, 2004.
14. M. Naor and O. Reingold. A pseudo-random encryption mode. Manuscript, available from www.wisdom.weizmann.ac.il/∼naor/.
15. M. Naor and O. Reingold. On the construction of pseudo-random permutations: Luby-Rackoff revisited. *Journal of Cryptology*, 12(1):29–66, 1999. (Earlier version in STOC '97.) Available from www.wisdom.weizmann.ac.il/∼naor/.
16. J. Patarin. Luby-Rackoff: 7 rounds are enough for $2^{n(1-\varepsilon)}$ security. In *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 513–529. Springer-Verlag, 2003.
17. S. Patel, Z. Ramzan and G .Sundaram. Efficient Constructions of Variable-Input-Length Block Ciphers. In Selected Areas in Cryptography – SAC 2004, volume 3xxx of *Lecture Notes in Computer Science*, pages yyy–zzz. Springer-Verlag, 2004.
18. R. L. Rivest. The RC5 encryption algorithm. In *Fast Software Encryption (FSE '94)*, volume 1008 of *Lecture Notes in Computer Science*, pages 86–96. Springer, 1994.
19. P. Rogaway. Efficient instantiations of tweakable block ciphers and refinements to modes OCB and PMAC. Available on-line from `http://www.cs.ucdavis.edu/∼rogaway/papers/`, 2004.

A Some Aspects of the Security-Proof

The proof of Theorem 1 is very similar to the security proof of EME [8]. We begin by viewing an attack against EME* as a game between the attacker and the mode itself, where the cipher is replaced by a truly random permutation π and this permutation is chosen "on the fly" during this game. We give names to all of the internal blocks that occur in the game, where an internal block is any of the n-bit values PP_i, PPP_i, MP_j, MC_j, MM, CCC_i, CC_i that arise as the game is played. For example, PPP_i^s is the PPP_i-block of the s^{th} query of the attacker.

The heart of the proof is to show that "accidental collisions" are unlikely. An accidental collision is an equality between two internal blocks that is not obviously guaranteed due to the structure of the mode. Specifically, an equality between the i^{th} blocks in two different encipher queries $P_i^s = P_i^t$ implies that we also have the equalities $PP_i^s = PP_i^t$ and $PPP_i^s = PPP_i^t$ and so these do not count as collisions. (And likewise for decipher queries.) Most other collisions are considered accidental collisions and we show that those rarely happen. Showing that accidental collisions are rare is ultimately done by case analysis. For example, in one case we show that with high probability $PP_i^s \neq PP_{i'}^t$; in another case

we show that with high probability $PPP_i^s \neq MC_j^t$, etc. Before we can get to the case analysis, however, we have to go through a game-substitution sequence in which the original scenario of an interactive attack is simplified and the interaction is removed. One aspect in this proof which is absent from the proof of EME is an analysis of the hashing $H_{K,R}(T)$, which is described below. The other aspects are similar to the original proof (although we need to handle more cases in the case analysis). The reader is refereed to the foll version [9] for these details.

Abstracting the function $H_{K,R}$: We somewhat simplify the analysis by replacing the function $H_{K,R}$ by an abstract function $h : \{0,1\}^* \to \{0,1\}^n$, chosen from a pairwise independent family $\mathcal{H}$. The properties of h that we use in the analysis are:

(i) For a fixed $\mathsf{T} \in \{0,1\}^*$, $h(\mathsf{T})$ is uniform in $\{0,1\}^n$ when h is chosen at random from $\mathcal{H}$.

(ii) For fixed $\mathsf{T} \neq \mathsf{T}' \in \{0,1\}^*$, $h(\mathsf{T}) \oplus h(\mathsf{T}')$ is uniform in $\{0,1\}^n$ when $h \xleftarrow{\$} \mathcal{H}$.

(iii) The choice $h \xleftarrow{\$} \mathcal{H}$ is independent of all the other random choices in the "attack game".

We can justify these assumptions on h by replacing the computation of $E_K(T \oplus jR) \oplus jR$ (with j a constant) in lines 10, 11, and 12 of Figure 1, by the computation $f_j(T)$ where for each j we have an independent random function $f_j : \{0,1\}^n \to \{0,1\}^n$. It is known that replacing a masked random permutation by a collection of random functions this way entails only a negligible difference on the view of the adversary. Specifically, one could prove the following: Fix an adversary with three oracles $A^{E(\cdot),D(\cdot),F(\cdot,\cdot)}$, and consider the two following experiments.

- In the first experiment (Expr1), we choose at random a permutation π over $\{0,1\}^n$ and a string $R \in \{0,1\}^n$. Then for $x, y, j \in \{0,1\}^n$ with $j \neq 0$, an oracle-query $E(x)$ is answered by $\pi(x)$, an oracle query $D(y)$ is answered by $\pi^{-1}(y)$, and an oracle query $F(j,x)$ is answered by $\pi(x \oplus jR) \oplus jR$ (where the multiplication jR is over $GF(2^n)$).
- In the second experiment (Expr2), we choose at random a permutation π over $\{0,1\}^n$, and 2^n functions $\{f_j : \{0,1\}^n \to \{0,1\}^n\}_{j \in \{0,1\}^n}$. Then for $x, y, j \in \{0,1\}^n$, with $j \neq 0$, the oracle-queries $E(x)$ and $D(y)$ are answered as before by $\pi(x)$ and $\pi^{-1}(y)$, respectively, but an oracle query $F(j,x)$ is answered by $f_j(x)$.

Lemma 1. *Fix some $n, q_p, q_f \in \mathbb{N}$. For any adversary $A^{E(\cdot),D(\cdot),F(\cdot,\cdot)}$ as above that makes at most q_p queries to E and D, and at most q_f queries to F, it holds that*

$$\left| \Pr_{\text{Expr1}}[A^{E,D,F} \Rightarrow 1] - \Pr_{\text{Expr2}}[A^{E,D,F} \Rightarrow 1] \right| \leq q_f(q_f + 2q_p)/2^n \qquad \square$$

This lemma is pretty much folklore by now, although I could not find a reference where it is proven. A similar result was proven by by Even and Mansour [5]

(but the masks there are completely independent, rather than "pairwise independent"). A proof for a special case of this lemma can be found in [1–Lemma 4], and that proof can easily be extended to prove Lemma 1 itself.

Using Lemma 1, we can replace the function $H_{K,R}$ from Figure 1 by the following function h (that depends on the 2^n random functions f_j). In the code below, the constants 2^i are computed in the finite field $GF(2^n)$.

function $h(T_1 \cdots T_{\ell-1}, T_\ell)$: // $|T_1| = \cdots = |T_{\ell-1}| = n$, $0 < |T_\ell| \le n$

01 **if** T is empty **return** $f_1(0)$

10 **for** $i \in [1..\ell-1]$ **do** $TTT_i \leftarrow f_{2^i}(T_i)$
11 **if** $|T_\ell| = n$ **then** $TTT_\ell \leftarrow f_{2^\ell}(T_\ell)$
12 **else** $TTT_\ell \leftarrow f_{2^{\ell+1}}(T_\ell 10..0))$
13 **return** $TTT_1 \oplus \cdots \oplus TTT_\ell$

Going back to the analysis of EME*, let $N_{\rm be}$ denote the *total number of block encryptions* that are used throughout the attack *not counting the computation of* H, and we clearly have

$$N_{\rm be} < (2 + \frac{1}{n})\sigma_n^d + 2q \tag{2}$$

Then from Lemma 1 it follows that the statistical distance in the view of the adversary due to the replacement of $H_{K,R}$ by h is bounded by $\sigma_n^a(\sigma_n^a + 2N_{\rm be})/2^n$. Once we made that replacement, it is clear that the choice of h is now independent of all the other random choices in the attack, so we only need to prove the properties (i) and (ii). This is done next:

Claim. When 2^n functions $\{f_j : \{0,1\}^n \to \{0,1\}^n\}_{j\in\{0,1\}^n}$ are chosen at random and h is defined as above, it holds that:
(i) For fixed $T \in \{0,1\}^{0..n(2^n-3)}$, $h(T)$ is uniform in $\{0,1\}^n$.
(ii) For fixed $T \neq T' \in \{0,1\}^{0..n(2^n-3)}$, $h(T) \oplus h(T')$ is uniform in $\{0,1\}^n$.

Proof. Property (i) is obvious, since the output of h at any point T depend on at least one application of one of the functions f_j, and these are all random functions. To prove Property (ii), fix some $T \neq T'$, and denote $T = T_1 \ldots T_\ell$ and similarly $T' = T'_1 \ldots T'_{\ell'}$, where $\ell = \lceil |T|/n \rceil$ and $\ell' = \lceil |T'|/n \rceil$. (The proof below uses the fact that 2 is a primitive element in $GF(2^n)$ and $\ell' \le 2^n - 3$, so for any $i, i' \le \ell' + 1$ if $i \neq i'$ then also $2^i \neq 2^{i'}$ in $GF(2^n)$.)

If $\ell = \ell'$ then there must be at least one index $i \le \ell$ such that $T_i \neq T'_i$. If T_i and T'_i are full blocks then

$$h(T) \oplus h(T') = \text{something-independent-of-}f_{2^i} \oplus f_{2^i}(T_i) \oplus f_{2^i}(T'_i),$$

which is uniform since f_{2^i} is a random function. If T_i, T'_i are both partial blocks (so $i = \ell$) then we get

$$h(T) \oplus h(T') = \text{something-independent-of-}f_{2^{\ell+1}} \oplus f_{2^{\ell+1}}(T_i 10..0) \oplus f_{2^{\ell+1}}(T'_i 10..0),$$

which is again uniform since $T_i \neq T'_i$ implies that also $T_i10..0 \neq T'_i10..0$ and $f_{2^{\ell+1}}$ is a random function. If T_i is a full block and T'_i is partial, then we similarly get $h(T) \oplus h(T') =$ something-independent-of-$f_{2^{\ell+1}} \oplus f_{2^{\ell+1}}(T'_i10..0)$.

If $\ell \neq \ell'$, then assume that $\ell' > \ell$. If T'_i is a partial block then as before we get $h(T) \oplus h(T') =$ something-independent-of-$f_{2^{\ell'+1}} \oplus f_{2^{\ell'+1}}(T'_i10..0)$. Similarly if T'_i is a full block and either $\ell' > \ell+1$ or T_ℓ is a full block, then $h(T) \oplus h(T') =$ something-independent-of-$f_{2^{\ell'}} \oplus f_{2^{\ell'}}(T'_i)$. The last case is when $\ell' = \ell + 1$ and $T'_{\ell'}$ is a full block and T_ℓ is a partial block. In this case $h(T')$ includes the term $f_{2^\ell}(T'_\ell)$ but $h(T)$ is independent of f_{2^ℓ}, so again $h(T) \oplus h(T')$ is uniform.

Impossibility of Construction of OWHF and UOWHF from PGV Model Based on Block Cipher Secure Against ACPCA

Donghoon Chang[1], Wonil Lee[1], Seokhie Hong[1], Jaechul Sung[2], Sangjin Lee[1], and Soohak Sung[3]

[1] Center for Information and Security Technologies, Korea University, Seoul, Korea
{dhchang, wonil, hsh, sangjin}@cist.korea.ac.kr
[2] Department of Mathematics, University of Seoul, Korea
jcsung@uos.ac.kr
[3] Applied Math. Department, Paichai University, Daejeon, Korea
sungsh@mail.paichai.ac.kr

Abstract. In 1993, Preneel, Govaerts and Vandewalle [11] considered 64 block cipher based hash functions (64 PGV-hash functions). In 2002, Black, Rogaway and Shrimpton [3] proved that 20 of 64 PGV-hash functions are collision resistant, assumed that a block cipher is a random block cipher. In 2002, Hirose [4] defined ACPA(Adaptive Chosen Plaintext Attack) model and ACPCA(Adaptive Chosen Plaintext/Ciphertext Attack) model and he showed that, for every PGV-hash function, there exist block ciphers secure against ACPA such that the PGV-hash function based on them is not a OWHF which has the properties of preimage resistance and second-preimage resistance. Recently, Lee *et al.* [6] generalized the definition of PGV-hash function into a hash family and showed that 42 of 64 PGV-hash families are collision resistant. In this paper, we show that, for every PGV-hash function, there exist block ciphers secure against ACPCA such that the PGV-hash family based on them is not a OWHF. We also show that, for every PGV-hash family, there exist block ciphers secure against ACPCA such that the PGV-hash family based on them is not a UOWHF.

1 Introduction

Simon [16] showed that there always exists an efficient adversary which finds collisions in any hash function based on a block cipher if we assume only that the block cipher is a pseudo-random permutation. So, all the proofs of collision resistance of block cipher-based hash functions have been done in the black-box model (instead of a general model such that a block cipher is pseudorandom) in which a block cipher is treated as a random and independent permutation for each key. In 2002, Black, Rogaway and Shrimpton [3] proved that 20 of 64 PGV-hash functions are collision resistant in the black-box model. Hirose [4]

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 328–342, 2004.

pointed out that the black-box model is impractical in a strict sense and considered other model such as the adaptive chosen plaintext attack(ACPA) and the adaptive chosen plaintext/ciphertext attack(ACPCA) model. He also showed that, for every PGV-hash function (PGV-hash function is a special case of PGV-hash family. See the section 2.), there exist block ciphers secure in ACPA model such that the PGV-hash function based on them doesn't have the property of one-wayness(preimage resistance + second-preimage resistance). But, in case of ACPCA model, he showed that 58 PGV-hash functions except with 6 cases are not secure construction for one-wayness (5 cases are not preimage resistant and 53 cases are not second-premiage resistant) and remained 6 cases as open problems in the conclusion section. Recently, Lee *et al.* [6] generalized the definition of PGV-hash function into a hash family and showed, using a proof technique which is simpler than that of [3], 42 of 64 PGV-hash families are collision resistant in the black-box model assuming that key length l is big. The result of [6] also implies that 42 collision resistant PGV-hash families are UOWHFs. Lee *et al.* do not use any mask key unlike [2, 5, 7, 8, 9, 10, 13, 14, 15] whose demerit is that the key length needed increases according to the message length. I.e, Lee *et al.*' result firstly shows that it is possible to construct a UOWHF only using a fixed length key in the black-box model.

Contribution of This Paper. This paper shows that there is no OWHF and no UOWHF in 64 PGV-hash functions and 64 PGV-hash families respectively if we assume only that the block cipher is secure against ACPCA. This means that the secure block cipher in ACPCA model is not a sufficient condition for a OWHF and a UOWHF constructed from PGV-hash function and PGV-hash family, respectively. Therefore, we need more conditions for constructing OWHFs and UOWHFs from 64 PGV model. But it is an open problem what conditions are needed at least. And the result of this paper is not the simple extension of Hirose's proof and an independent work by four reasons. First, Hirose did not show whether there exist block ciphers secure against ACPCA such that 6 PGV-hash families based on them aren't a OWHF. Second, a second-preimage resistant hash function(a SPRHF) considered by Hirose is not related to a UOWHF. In FSE 2004, Rogaway and Shrimpton [12] defined the 7 security notions of the compression function and studied the relations between these 7 notions. According to these notions, a SPRHF considered by Hirose corresponds to the aSec and a UOWHF corresponds to eSec and these two notions have no relation. Third, results of [12] can not be used to show the relation between SPRHF considered by Hirose and a UOWHF because [12] compared the security notions with using a fixed compression function family but compression functions considered by Hirose and Lee *et al.* are different. (Hirose lets the length of the input message 'n'. On the other hand, Lee *et al.* let '$n - l$'.) Hirose also considered a hash function but Lee *et al.* considered a hash function family. Fourth, Hirose showed that 53 of 64 PGV-hash functions are not SPRHFs in the ACPCA model.

Therefore, we can know that the study about OWHFs and UOWHFs from 64 PGV-hash function and 64 PGV-hash families in ACPCA model is necessary and independent from Hirose's work. In this paper, we improve the method of

Hirose's proof and show that, for every PGV-hash function and every PGV-hash family, there exist block ciphers secure against ACPCA such that the PGV-hash fuction and PGV-hash family based on them is not a OWHF and a UOWHF, respectively.

The organization of this paper is as follows. In section 2, we define the notations. And then, in section 3, we show that there is no construction of UOWHF from 64 PGV-hash families in the ACPCA model. In section 4, we show that there is no construction of OWHF from 64 PGV-hash functions in the ACPCA model. Finally, we conclude and suggest a future work.

2 Preliminaries

Basic Notations

1. A *block cipher* is a map $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ where, for each key $a \in \{0,1\}^n$, the function $E_a(\cdot) = E(a,\cdot)$ is a permutation on $\{0,1\}^n$. If E is a block cipher then E^{-1} is its inverse, where $E_a^{-1}(y)$ is the string x such that $E_a(x) = y$.
2. A *hash family* is a $\mathcal{H} = \{H_k\}_{k\in\{0,1\}^l}$, where $H_k : (\{0,1\}^{n-l})^* \to \{0,1\}^n$.

General Definition Of PGV-Hash Family [6]. Let $0 \le l < n$ and $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher. If $l = 0$ let $\{0,1\}^0 = \{\epsilon\}$, where ϵ is the empty string. Using the block cipher E, we want to construct a compression function family $\mathcal{F} = \{f^k\}_{k\in\{0,1\}^l}$, $f^k : \{0,1\}^n \times \{0,1\}^{n-l} \to \{0,1\}^n$.

Let $h_0, v \in \{0,1\}^n$ be fixed values. We define the 64 ways to construct a (*block-cipher-based*) *compression function family* $\mathcal{F} = \{f^k\}_{k\in\{0,1\}^l}$ in the following manner: for each $k \in \{0,1\}^l$,

$$f^k(h,m) = E_a(b) \oplus c,$$

where $a, b, c \in \{h, (m||k), h \oplus (m||k), v\}$. Note that $|h| = n$ and $|m| = n - l$. Then we can define the PGV-hash family $\mathcal{H} = \{H^k\}_{k\in\{0,1\}^l}$ from the compression function family $\mathcal{F} = \{f^k\}_{k\in\{0,1\}^l}$ as follows: for each $k \in \{0,1\}^l$, $H^k : (\{0,1\}^{n-l})^* \to \{0,1\}^n$ is defined by

function $H^k(m_1 \cdots m_t)$
for $i \leftarrow 1$ **to** t **do** $h_i \leftarrow f^k(h_{i-1}, m_i)$
return h_t

Note that the key k of PGV-hash family is equal to the key of compression function family and note that if $l = 0$ then $\mathcal{F} = \{f^k\}_{k\in\{0,1\}^0} = \{f^\epsilon\}$ is a singleton set and this is corresponding to the original definition of PGV [11].

Second-Preimage Resistance of a PGV-hash function ($l = 0$ in case of a PGV-hash family). We quantify second-preimage resistance of a (block-cipher-based) hash function $\mathcal{H}$. Then, the adversary $\mathcal{A}$ for second-preimage resistance plays the following game called *Spi*.

1. An M is given randomly.
2. $\mathcal{A}$ has to find M' such that $M \neq M'$ but $\mathcal{H}(M) = \mathcal{H}(M')$.

Definition 1. *(Second-Preimage Resistance of a PGV-hash function '$\mathcal{H}$') Let $\mathcal{H}$ be a PGV-hash function, where $\mathcal{H} : (\{0,1\}^n)^* \to \{0,1\}^n$. Then the advantage of $\mathcal{A}$ with respect to (target) second-preimage resistance are the the following real numbers.*

$$\mathbf{Adv}_{\mathcal{H}}^{Spi}(\mathcal{A}) = Pr[M \xleftarrow{R} \{0,1\}^*; M' \leftarrow \mathcal{A}(M) : M \neq M' \ \& \ \mathcal{H}(M) = \mathcal{H}(M')]$$

When the maximum running time of the adversary is t, the maximum advantage is defined as follow.

$$\mathbf{Adv}_{\mathcal{H}}^{Spi}(t) = Max_{\mathcal{A}}\{\mathbf{Adv}_{\mathcal{H}}^{Spi}(\mathcal{A})\}$$

Target Collision Resistance Of Hash Family $(0 < l < n)$ The notion UOWHF is introduced by Naor and Yung [10]. Bellare and Rogaway used the target collision resistance as the non-asymptotic version for the asymptotic property of UOWHF [2]. We quantify target collision resistance of a (block-cipher-based) hash family $\{H^k\}_{k\in\{0,1\}^l}$. Then, the adversary $\mathcal{A} = (\mathcal{A}_{guess}, \mathcal{A}_{find}(\cdot,\cdot))$ for target collision resistance plays the following game called $TColl$.

1. $\mathcal{A}_{guess}$ commits to an M.
2. A key k is chosen uniformly at random from $\{0,1\}^l$.
3. $\mathcal{A}_{find}(M,k)$ has to find M' such that $M \neq M'$ but $H_k(M) = H_k(M')$.

Definition 2. *(Target collision resistance of a PGV-hash family '$\mathcal{H}$') Let $\mathcal{H} = \{H_k\}_{k\in\{0,1\}^l}$ be a PGV-hash family, where $H_k : (\{0,1\}^{n-l})^* \to \{0,1\}^n$. Then the advantage of $\mathcal{A}$ with respect to (target) collision resistance are the the following real numbers.*

$$\mathbf{Adv}_{\mathcal{H}}^{TColl}(\mathcal{A}) = Pr[M \leftarrow \mathcal{A}_{guess}; k \xleftarrow{R} \{0,1\}^l; M' \leftarrow \mathcal{A}_{find}(M,k) : M \neq M' \ \& \ H_k(M) = H_k(M')]$$

When the maximum running time of the adversary is t, the maximum advantage is defined as follow.

$$\mathbf{Adv}_{\mathcal{H}}^{TColl}(t) = Max_{\mathcal{A}}\{\mathbf{Adv}_{\mathcal{H}}^{TColl}(\mathcal{A})\}$$

Security Notion of a Block Cipher Considered in This Paper. Find-then-Guess security [1] is well known as the security notion of the symmetric encryption schemes. In [4], the security of a block cipher in ACPA and ACPCA model is considered. This security is similar to Find-then-Guess Security. Therefore, we express the experiment in ACPA and ACPCA model into FtG-ACPA and FtG-ACPCA. We imagine an adversary that runs in two stages. During the find stage, the adversary is given access to oracles $E_a(\cdot), E_a^{-1}(\cdot)$ and oracle queries

are q at most. He endeavors to come up with (x^0, x^1, r, s), whose encryptions(r=1) or decryptions(r=-1) it wants to try to tell apart. It also retains some state information s that it may want to preserve to help it later. In the guess stage, it is given a random ciphertext(case r=1) or a random plaintext(case r=-1) y for one of x^0, x^1, together with the state information s. The adversary "wins" if it correctly identifies which one of x^0, x^1 goes with y. The block cipher is "good" if "reasonable" adversaries cannot win significantly more than half the time. In this paper we informally use the term "secure" instead of "good".

Definition 3. *[FtG-ACPA, FtG-ACPCA] Let E be a block cipher. Let $b \in \{0,1\}$ and $r \in \{1,-1\}$ Let $\mathcal{A}$ be an adversary. Now, we consider the following experiments:*

Experiment $\mathsf{Exp}_{E,\mathcal{A}}^{FtG-ACPA-b}(q)$	Experiment $\mathsf{Exp}_{E,E^{-1},\mathcal{A}}^{FtG-ACPCA-b}(q)$
$a \xleftarrow{R} \{0,1\}^n$	$a \xleftarrow{R} \{0,1\}^n$
$(x^0, x^1, s) \leftarrow \mathcal{A}^{E_a(\cdot)}(\mathsf{find})$	$(x^0, x^1, r, s) \leftarrow \mathcal{A}^{E_a(\cdot), {E_a}^{-1}(\cdot)}(\mathsf{find})$
$y \leftarrow E_a(x^b)$	$y \leftarrow {E_a}^r(x^b)$
$b' \leftarrow \mathcal{A}(\mathsf{guess}, y, s)$	$b' \leftarrow \mathcal{A}(\mathsf{guess}, r, y, s)$
Return b'	Return b'

It is mandated that $\mathcal{A}$ in Experiment FtG-ACPA chooses x^0, x^1 which he doesn't ask in find stage and that $\mathcal{A}$ in Experiment FtG-ACPCA chooses x^0, x^1, r such that he doesn't ask ${E_a}^r(x^0)$ and ${E_a}^r(x^1)$ in find stage.

The advantage of the adversary $\mathcal{A}$ is defined as follow.

$$\mathbf{Adv}_{E,\mathcal{A}}^{FtG-ACPA}(q) = |Pr[\mathsf{Exp}_{E,\mathcal{A}}^{FtG-ACPA-1}(q) = 1] - Pr[\mathsf{Exp}_{E,A}^{FtG-ACPA-0}(q) = 1]|$$

$$\mathbf{Adv}_{E,E^{-1},\mathcal{A}}^{FtG-ACPCA}(q) = |Pr[\mathsf{Exp}_{E,E^{-1},\mathcal{A}}^{FtG-ACPCA-1}(q) = 1] - Pr[\mathsf{Exp}_{E,E^{-1},A}^{FtG-ACPCA-0}(q) = 1]|$$

the maximum advantage is defined as follow.

$$\mathbf{Adv}_{E}^{FtG-ACPA}(q) = Max_{\mathcal{A}}\{\mathbf{Adv}_{E,\mathcal{A}}^{FtG-ACPA}(q)\}$$

$$\mathbf{Adv}_{E,E^{-1}}^{FtG-ACPCA}(q) = Max_{\mathcal{A}}\{\mathbf{Adv}_{E,E^{-1},\mathcal{A}}^{FtG-ACPCA}(q)\}$$

3 Constructions to Secure Block Cipher Against ACPCA Such That PGV-Hash Families Based on Them Are Not Target Collision Resistant

This section suggests constructions to secure block cipher against ACPCA which is not target collision resistant in PGV-model. First, we suppose that E^* is a secure block cipher against ACPCA. Then, we construct E's in case of [E-1]∼[E-18] as follows. E^* use a $n-1$ bit key in [E-15,16,18] and $n-2$ bit key in [E-17] and n bit key in other cases.

[E-1]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,
else if $a[low\ l]=0^l$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-2]If $a=0^n$ or $a=h_0$, then $E_a(x)=x$,
else $E_a(x)=E_a^*(x)$

[E-3]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x\oplus h_0$,
else if $a[low\ l]=0^l$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-4]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x\oplus a$,
else if $a[low\ l]=0^l$, then $E_a(x)=x$,
else $E_a(x)=E_a^*(x)$

[E-5]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,
else if $a[low\ l]=h_0[low\ l]$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-6]If $a=h_0$, then $E_a(x)=x$,
else if $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,
else if $a[low\ l]=0^l$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-7]If $a=h_0$, then $E_a(x)=x\oplus a$,
else if $a[low\ l]=0^l$, then $E_a(x)=x$,
else if a[high n-l]=h_0[high n-l], then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-8]If $a=h_0$, then $E_a(x)=x$,
else if $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x\oplus a$,
else if $a[low\ l]=0^l$, then $E_a(x)=x$,
else $E_a(x)=E_a^*(x)$

[E-9]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,
else if $a[low\ l]=0^l$, then $E_a(x)=x$,
else $E_a(x)=E_a^*(x)$

[E-10]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x\oplus a$,
else if $a[low\ l]=0^l$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-11]If $a=h_0$, then $E_a(x)=x\oplus a$,
else if $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,
else if $a[low\ l]=0^l$, then $E_a(x)=x\oplus a$,
else $E_a(x)=E_a^*(x)$

[E-12]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x$,

$$otherwise\ E_a(x)=\begin{cases} a & if\ x=a \\ E_a^*(a) & if\ x=E_a^{*-1}(a) \\ E_a^*(x) & otherwise \end{cases}$$

[E-13]If $a[high\ n-l]=0^{n-l}$, then $E_a(x)=x\oplus h_0$,

$$otherwise\ E_a(x)=\begin{cases} a & if\ x=a \\ E_a^*(a) & if\ x=E_a^{*-1}(a) \\ E_a^*(x) & otherwise \end{cases}$$

[E-14]

$$E_a(x)=\begin{cases} x & if\ x=a\ or\ a\oplus h_0 \\ E_a^*(E_a^*(x)) & if x=E_a^{*-1}(a) or E_a^{*-1}(a\oplus h_0) \\ E_a^*(x) & otherwise \end{cases}$$

[E-15]If $a=h_0$, then $E_a(x)=x$,
else $E_a(x)=E^*_{a[low\ n-1]}(x)$,

[E-16]

$$E_a(x)=\begin{cases} if\ a[high\ 1]\neq x[high\ 1] \\ \quad E^*_{a[low\ n-1]}(1||x[low\ n-1]) \\ if\ a[high\ 1]=x[high\ 1] \\ \quad E^*_{a[low\ n-1]}(0||x[low\ n-1]) \end{cases}$$

[E-17]If $a[high\ 2th\ bit]=1$, $E_a(x)=E^*_{a[low\ n-2]}(x)$,
If $a[high\ 2th\ bit]=0$,

$$E_a(x)=\begin{cases} if\ a[high\ 1]\neq x[high\ 1] \\ \quad E^*_{a[low\ n-2]}(1||x[low\ n-1]) \\ if\ a[high\ 1]=x[high\ 1] \\ \quad E^*_{a[low\ n-2]}(0||x[low\ n-1]) \end{cases}$$

[E-18]

$$E_a(x)=\begin{cases} E^*_{a[low\ n-1]}(x)\oplus 10^{n-1} & if\ a[high\ 1]=0 \\ E^*_{a[low\ n-1]}(x) & if\ a[high\ 1]=1 \end{cases}$$

a is a n-bit key and x is any n-bit input. $a[high\ n-l]$ means the leftmost $n-l$ bits of a and $a[low\ l]$ means the rightmost l bits of a. 0^{n-l} is an all zero-bit string such that $|0^{n-l}|=n-l$. Then, E's in case of [E-1]∼[E-18] are secure in ACPCA model if E^* is secure in ACPCA model and $n-l$ is big and l is big.

Theorem 1. *For [E-1]∼[E-18], block ciphers E's constructed with secure block ciphers E^*'s against ACPCA are also secure against ACPCA.*

Here, we show only a proof in case of [E-16]. For other cases, see the appendix A. For [E-16],

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q)\leq \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q)$$

Proof. Using an adversary $\mathcal{A}$ making ACPCA for E, we construct an adversary $\mathcal{C}_{\mathcal{A}}$ making ACPCA for $E^*=\{e^{*i}\}_{i\in\mathbb{N}}$ in [E-16] as follows.

► Adversary $\mathcal{C}_{\mathcal{A}}$

$\mathcal{C}_{\mathcal{A}}^{E_a^*(\cdot),E_a^{*-1}}(find)$: Choose $w\in\{0,1\}$ randomly. Then run $\mathcal{A}^{E_a(\cdot),E_a^{-1}}(find)$. $\mathcal{C}_{\mathcal{A}}$ simulates $\mathcal{A}$'s oracles $E_a(\cdot)$ and $E_a^{-1}(\cdot)$ with $\mathcal{C}_{\mathcal{A}}$'s oracles $E_a^*(\cdot)$ and $E_a^{*-1}(\cdot)$. For each $\mathcal{A}$'s query (x_i,r_i), in case of r_i=1, if $w\neq x_i[high\ 1]$ then $\mathcal{C}_{\mathcal{A}}$ gives $1||x_i[low\ n-1]$ to oracle $E_{a'}^*(\cdot)$, if $w=x_i[high\ 1]$ then $\mathcal{C}_{\mathcal{A}}$ gives $0||x_i[low\ n-1]$ to oracle $E_{a'}^*(\cdot)$. Then $\mathcal{C}_{\mathcal{A}}$ gives the value y_i obtained from oracle $E_{a'}^*(\cdot)$ to $\mathcal{A}$. In case of r_i=-1, $\mathcal{C}_{\mathcal{A}}$ gives x_i to oracle ${E_{a'}^*}^{-1}(\cdot)$ and obtains $y_i={E_{a'}^*}^{-1}(x_i)$ from oracle ${E_{a'}^*}^{-1}(\cdot)$. If $y_i[high\ 1]=1$, $\mathcal{C}_{\mathcal{A}}$ gives $w\oplus 1||y[low\ n-1]$ to $\mathcal{A}$, if $y_i[high\ 1]=0$, $\mathcal{C}_{\mathcal{A}}$ gives $w||y[low\ n-1]$ to $\mathcal{A}$. At last, $\mathcal{A}^{E_a(\cdot),{E_a}^{-1}}(find)$ outputs (x^0,x^1,r). If $r=1$, after $\mathcal{C}_{\mathcal{A}}$ checks whether $w\neq x^t[high\ 1]$

Case	PGV-Hash Family	Block Cipher	M	M'
1	$E_{x_i}(x_i) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
2	$E_{h_{i-1}}(x_i) \oplus v$	E-15	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
3	$E_{w_i}(x_i) \oplus v$	E-1	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
4	$E_v(x_i) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
5	$E_{x_i}(x_i) \oplus x_i$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
6	$E_{h_{i-1}}(x_i) \oplus x_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
7	$E_{w_i}(x_i) \oplus x_i$	E-12	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
8	$E_v(x_i) \oplus x_i$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
9	$E_{x_i}(x_i) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
10	$E_{h_{i-1}}(x_i) \oplus h_{i-1}$	E-15	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2 \oplus 10^{n-l-1}$
11	$E_{w_i}(x_i) \oplus h_{i-1}$	E-3	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
12	$E_v(x_i) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
13	$E_{x_i}(x_i) \oplus w_i$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
14	$E_{h_{i-1}}(x_i) \oplus w_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
15	$E_{w_i}(x_i) \oplus w_i$	E-13	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
16	$E_v(x_i) \oplus w_i$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
17	$E_{x_i}(h_{i-1}) \oplus v$	E-15	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
18	$E_{h_{i-1}}(h_{i-1}) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
19	$E_{w_i}(h_{i-1}) \oplus v$	E-4	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
20	$E_v(h_{i-1}) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
21	$E_{x_i}(h_{i-1}) \oplus x_i$	E-18	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
22	$E_{h_{i-1}}(h_{i-1}) \oplus x_i$	E-2	$m_1 \| \| m_1$	$m_1' \| \| m_1'$
23	$E_{w_i}(h_{i-1}) \oplus x_i$	E-5	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
24	$E_v(h_{i-1}) \oplus x_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_1 \oplus m_2 \oplus m_1'$
25	$E_{x_i}(h_{i-1}) \oplus h_{i-1}$	E-15	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
26	$E_{h_{i-1}}(h_{i-1}) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
27	$E_{w_i}(h_{i-1}) \oplus h_{i-1}$	E-4	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
28	$E_v(h_{i-1}) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
29	$E_{x_i}(h_{i-1}) \oplus w_i$	E-17	$(11m_1) \| \| (10m_2), \|m_i\| = n-l-2$	$(01m_1) \| \| (00m_2), \|m_i\| = n-l-2$
30	$E_{h_{i-1}}(h_{i-1}) \oplus w_i$	E-12	$m_1 \| \| m_2$	$m_1' \| \| m_2$
31	$E_{w_i}(h_{i-1}) \oplus w_i$	E-1	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
32	$E_v(h_{i-1}) \oplus w_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2$
33	$E_{x_i}(w_i) \oplus v$	E-16	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
34	$E_{h_{i-1}}(w_i) \oplus v$	E-6	$0 \| \| m_2 \| \| m_3$	$0 \| \| m_2' \| \| m_3$
35	$E_{w_i}(w_i) \oplus v$	E-5	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
36	$E_v(w_i) \oplus v$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_1 \oplus m_2 \oplus m_1'$
37	$E_{x_i}(w_i) \oplus x_i$	E-13	$0 \| \| m_2$	$0 \| \| m_2'$
38	$E_{h_{i-1}}(w_i) \oplus x_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
39	$E_{w_i}(w_i) \oplus x_i$	E-13	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
40	$E_v(w_i) \oplus x_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
41	$E_{x_i}(w_i) \oplus h_{i-1}$	E-16	$m_1 \| \| m_2$	$m_1 \oplus 10^{n-l-1} \| \| m_2$
42	$E_{h_{i-1}}(w_i) \oplus h_{i-1}$	E-7	$h_0[high\ n-l] \| \| m_2 \| \| m_3$	$h_0[high\ n-l] \| \| m_2' \| \| m_3$
43	$E_{w_i}(w_i) \oplus h_{i-1}$	E-12	$m_1 \| \| m_2$	$m_1' \| \| m_2$
44	$E_v(w_i) \oplus h_{i-1}$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2$
45	$E_{x_i}(w_i) \oplus w_i$	E-14	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
46	$E_{h_{i-1}}(w_i) \oplus w_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2$
47	$E_{w_i}(w_i) \oplus w_i$	E-12	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
48	$E_v(w_i) \oplus w_i$	E-2	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
49	$E_{x_i}(v) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
50	$E_{h_{i-1}}(v) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
51	$E_{w_i}(v) \oplus v$	E-4	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
52	$E_v(v) \oplus v$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
53	$E_{x_i}(v) \oplus x_i$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
54	$E_{h_{i-1}}(v) \oplus x_i$	E-8	$0 \| \| m_2 \| \| m_3$	$0 \| \| m_2' \| \| m_3$
55	$E_{w_i}(v) \oplus x_i$	E-9	$h_0[high\ n-l] \| \| m_2 \| \| m_3$	$h_0[high\ n-l] \| \| m_2' \| \| m_3$
56	$E_v(v) \oplus x_i$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2$
57	$E_{x_i}(v) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
58	$E_{h_{i-1}}(v) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
59	$E_{w_i}(v) \oplus h_{i-1}$	E-10	$h_0[high\ n-l] \| \| m_2 \| \| m_3$	$h_0[high\ n-l] \| \| m_2' \| \| m_3$
60	$E_v(v) \oplus h_{i-1}$	Arbitrary	$m_1 \| \| m_2$	$m_1' \| \| m_2'$
61	$E_{x_i}(v) \oplus w_i$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$
62	$E_{h_{i-1}}(v) \oplus w_i$	E-11	$0 \| \| m_2 \| \| m_3$	$0 \| \| m_2' \| \| m_3$
63	$E_{w_i}(v) \oplus w_i$	E-1	$h_0[high\ n-l] \| \| m_2$	$h_0[high\ n-l] \| \| m_2'$
64	$E_v(v) \oplus w_i$	Arbitrary	$m_1 \| \| m_2$	$m_2 \| \| m_1$

Table 1. This table shows that all the 64 PGV-hash families are not secure construction for UOWHF. x_i means $m_i||k$. We write w_i for $x_i \oplus h_{i-1}$. Column 3 gives block ciphers in Section 3. M is chosen by an adversary and then randomly given key k, find M' such that $H_k\ (M)$=$H_k\ (M')$. For example, in case 3, 31 and 63, M and M' are systematically described at the proof of theorem 2

(t=0,1) for each x^0, x^1, $\mathcal{C}_\mathcal{A}$ gives (x^0, x^1, r) transformed by the previous same method to oracle $E^*_{a'}(\cdot)$. If $r = -1$, $\mathcal{C}_\mathcal{A}$ gives (x^0, x^1, r) to oracle ${E^*_{a'}}^{-1}(\cdot)$.

$\mathcal{C}_\mathcal{A}^{E^*_a(\cdot),{E^*_a}^{-1}}(guess) : \mathcal{C}_\mathcal{A}$ obtains $y = {E_a}^r(x^b)$. In case of $r = 1$, $\mathcal{C}_\mathcal{A}$ gives y to $\mathcal{A}$. In case of $r = -1$, if $y[high\ 1] = 1$ then $\mathcal{C}_\mathcal{A}$ gives $w \oplus 1||y[low\ n-1]$ to $\mathcal{A}$, if $y[high\ 1] = 0$ then $\mathcal{C}_\mathcal{A}$ gives $w||y[low\ n-1]$ to $\mathcal{A}$. At last $\mathcal{C}_\mathcal{A}$ outputs b' which $\mathcal{A}(guess, y, s)$ outputs.

Then,

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) = \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q),$$

Therefore, $$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q)$$ ■

Using E's (in case of [E-1]∼[E-18]) constructed above, we can show that all 64 PGV-hash families are not target collision resistant. I.e., we can find the target collisions for 64 PGV-hash families with probability 1.

Theorem 2. *64 PGV-hash families using block cipher E (which is constructed above) are not secure with respect to target collision resistance. I.e., there exist target collisions with probability 1 for every 64 PGV-hash families.*

Proof. Here, we show only a proof for case 3, 31 and 63 among 64 cases. For other cases, see the table 1.

[case 3,31,63]
1. Adversary $\mathcal{A}_{guess}$ commits to an $M = h_0[high\ n-l]||m_2$. Here, $h_0[high\ n-l]$ is the msb $n-l$ bits of initial value h_0.
2. A key $k \in \{0,1\}^n$ is chosen uniformly.
3. $\mathcal{A}_{find}(M, k)$ finds $M' = h_0[high\ n-l]||m'_2$. Here, m'_2 is any value such that $m'_2 \neq m_2$.
4. $H_k(M) = H_k(M')$ with probability 1. ■

4 Constructions to Secure Block Cipher Against ACPCA Such That PGV-Hash Functions Based on Them Are Not One-Way

Hirose [4] showed that 58 PGV-hash functions execpt 6 cases are not secure construction for one-wayness (5 cases are not preimage resistant and 53 cases are not second-premiage resistant) in ACPCA model. In this section, we show that other 6 PGV-hash functions are not secure construction for second-premiage resistance. I.e., all the 64 PGV-hash functions are not secure construction for one-wayness.

Table 2. This table shows that all the 64 PGV-hash functions (for other 58 cases, Hirose [4] showed) are not secure construction for OWHF. We write w_i for $m_i \oplus h_{i-1}$. Column 3 gives block ciphers in Section 3. M is given randomly and find M' such that $\mathcal{H}(M)=\mathcal{H}(M')$. For example, in case 25, M and M' are systematically described at the proof of theorem 3

Case	PGV-Hash Function	Block Cipher	M	M'
7	$E_{w_i}(m_i) \oplus m_i$	E-14	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$
14	$E_{h_{i-1}}(m_i) \oplus w_i$	E-2	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$
25	$E_{m_i}(h_{i-1}) \oplus h_{i-1}$	E-15	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$, $m'_i = m_i \oplus 10^{n-1}$
27	$E_{w_i}(h_{i-1}) \oplus h_{i-1}$	E-15	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$, $m'_i = m_i \oplus 10^{n-1}$
38	$E_{h_{i-1}}(w_i) \oplus m_i$	E-2	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$
45	$E_{m_i}(w_i) \oplus w_i$	E-14	$m_1 \| \cdots \| m_t$	$m'_1 \| \cdots \| m'_t$

Theorem 3. *64 PGV-hash functions are not secure constructions for one-wayness in ACPCA model.*

Proof. Hirose [4] already showed that 58 PGV-hash functions are not secure constructions for one-wayness in ACPCA model. So, we consider only 6 cases. Here, we show a proof for case 25 based on E in [E-15]. For other 5 cases, see the table 2.

[case 25] 1. An $M = m_1 \| \cdots \| m_t$ is given randomly.
2. $\mathcal{A}(M)$ chooses $M' = m'_1 \| \cdots \| m'_t$. Here, $m'_i = m_i \oplus 10^{n-1}$
3. $\mathcal{H}(M) = \mathcal{H}(M')$ with probability 1. ■

5 Conclusion

In this paper, we have considered ACPCA model instead of the black-box model and show the impossibility of secure construction of OWHF and UOWHF from PGV-hash functions and PGV-hash families in ACPCA model. It is an open problem whether there exists a model weaker than the black-box model such that PGV-hash functions and PGV-hash families are a OWHF and a UOWHF and a CRHF, assumed that a block cipher is secure against that model.

Acknowledgement

This work was supported (in part) by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program. The second author was supported by the 21st Century COE program 'Reconstruction of Social Infrastructure Related to Information Science and Electrical Engineering' of Kyushu University, Japan.

References

1. M. Bellare, A. Desai, E. Jokipii and P. Rogaway, *A Concrete Security Treatment of Symmetric Encryption*, Proc. the 38th IEEE FOCS, 1997.

2. M. Bellare and P. Rogaway, *Collision-resistant hashing: towards making UOWHFs practical*, Advances in Cryptology - CRYPTO'97, LNCS 1294, Springer-Verlag, pp. 470-484, 1997.
3. J. Black, P. Rogaway and T. Shrimpton, *Black-box analysis of the block-cipher-based hash function constructions from PGV*, Advances in Cryptology - CRYPTO'02, LNCS 2442, Springer-Verlag, pp. 320-335, 2002.
4. S. Hirose, *Secure Block Ciphers Are Not Sufficient for One-Way Hash Functions in the Preneel-Govaerts-Vandewalle Model*, SAC'02, LNCS 2595, Springer-Verlag, pp. 339-352, 2003.
5. W. Lee, D. Chang., S. Lee, S. Sung and M. Nandi, *New Parallel Domain Extenders for UOWHF*, In Asiacrypt'03, LNCS 2894, Springer-Verlag, pp. 208-227, 2003.
6. W. Lee, M. Nandi, P. Sarkar, D. Chang, S. Lee and K. Sakurai, *A Generalization of PGV-Hash Functions and Security Analysis in Black-Box Model*, In ACISP2004, LNCS 3108, Springer-Verlag, pp. 212-223, 2004.
7. I. Mironov, *Hash functions: from Merkle-Damgard to Shoup*, In Eurocrypt'01, LNCS 2045, Springer-Verlag, pp. 166-181, 2001.
8. M. Nandi, *A New Tree based Domain Extension of UOWHF*, Cryptology ePrint Archieve, http://eprint.iacr.org/2003/142.
9. M. Nandi, *Study of Domain Extention of UOWHF and its Optimaility*, Cryptology ePrint Archieve, http://eprint.iacr.org/2003/158.
10. M. Naor and M. Yung, *Universal one-way hash functions and their cryptographic applications*, Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing, ACM Press, pp. 33-43, 1989.
11. B. Preneel, R. Govaerts and J. Vandewalle, *Hash functions based on block ciphers:A synthetic approach*, Advances in Cryptology - CRYPTO'93, LNCS 773, Springer-Verlag, pp. 368-378, 1994.
12. P. Rogaway and T. Shrimpton, *Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance*, In FSE'04, LNCS 3017, Springer-Verlag, pp. 371-388, 2004.
13. P. Sarkar, *Construction of UOWHF: Tree Hashing Revisted*, Cryptology ePrint Archieve, http://eprint.iacr.org/2002/058.
14. P. Sarkar, *Domain Extenders for UOWHF: A Generic Lower Bound on Key Expansion and a Finite Binary Tree Algorithm* , Cryptology ePrint Archieve, http://eprint.iacr.org/2003/009.
15. V. Shoup, *A composition theorem for universal one-way hash functions*, In Eurocrypt'00, LNCS 1807, Springer-Verlag, pp. 445-452, 2000.
16. D. Simon. *Finding collisions on a one-way street: can secure hash functions be based on general assumptions?*, Advances in Cryptology - Eurocrypt'98, LNCS 1403, Springer-Verlag, pp. 334-345, 1998.

Appendix A : Proof of Theorem 1

1. Cases [E-1], [E-3], [E-4]∼[E-11], [E-2]

For [E-1],[E-3],[E-4]∼[E-11],

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q) + (2^{l-n+1} + 2^{1-l} - 2^{1-n})$$

For [E-2],

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q) + 2^{2-n}$$

Proof. $\divideontimes$ means the event that $E_a(\cdot) = E^*_a(\cdot)$ for randomly given key a.

[E-1], [E-3], [E-4]$\sim$[E-11]: $Pr[\divideontimes^c]$=$2^{l-n} + 2^{-l} - 2^{-n}$
[E-2]: $Pr[\divideontimes^c] = 2^{1-n}$

$\mathsf{Exp}(b) = 1$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E,E^{-1},\mathcal{A}}(q) = 1$ in the following inequalities.

$$\begin{aligned}
\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) &= |Pr[\mathsf{Exp}(1) = 1] - Pr[\mathsf{Exp}(0) = 1]| \\
&= |Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] + Pr[\mathsf{Exp}(1) = 1|\divideontimes^c]Pr[\divideontimes^c] \\
&\quad - Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0) = 1|\divideontimes^c]Pr[\divideontimes^c]| \\
&\leq |Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes]| \\
&\quad + |Pr[\mathsf{Exp}(1) = 1|\divideontimes^c]Pr[\divideontimes^c] - Pr[\mathsf{Exp}(0) = 1|\divideontimes^c]Pr[\divideontimes^c]| \\
&\leq |Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes]| + Pr[\divideontimes^c] \cdots (1) \\
&\leq |\mathsf{Exp}(1) = 1] - Pr[\mathsf{Exp}(1) = 1]| + 2 \cdot Pr[\divideontimes^c] \cdots (2) \\
&= \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{A}}(q) + 2 \cdot Pr[\divideontimes^c]
\end{aligned}$$

(1)$\rightarrow$(2) :

$$\begin{aligned}
Pr[\mathsf{Exp}(1) = 1] - Pr[\divideontimes^c] &\leq Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] \leq Pr[\mathsf{Exp}(1) = 1] \\
Pr[\mathsf{Exp}(0) = 1] - Pr[\divideontimes^c] &\leq Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes] \leq Pr[\mathsf{Exp}(0) = 1]
\end{aligned}$$

Above inequalities are checked easily. Here, let T=$Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes]$, T'=$Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes]$, ε=$Pr[\divideontimes^c]$, x=$Pr[\mathsf{Exp}(1) = 1]$, y=$Pr[\mathsf{Exp}(0) = 1]$. Then above inequalities are expressed as $x - \varepsilon \leq T \leq x$, $y - \varepsilon \leq T' \leq y$. Therefore, (1) $\rightarrow$ (2) holds because $|T - T'| \leq |x - y| + \varepsilon$. ■

2. Cases [E-12], [E-13]

For [E-12],[E-13],

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq (12 \cdot q + 1) \cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q) + (6 \cdot q + 4) \cdot 2^{l-n}$$

Proof. $\divideontimes$ means the event that $E_a(\cdot) = E^*_a(\cdot)$ for randomly given key a.

[E1-12], [E1-13]: $Pr[\divideontimes^c] = 2^{l-n}$

$\triangle$ means the event that there is no query x of an adversary $\mathcal{A}$ among maximum q queries such that $x \in \{a, E^{*-1}_a(a), E^*_a(a)\}$. $\mathsf{Exp}(b) = 1$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E,E^{-1},\mathcal{A}}(q) = 1$ in the following inequalities.

$$\begin{aligned}
\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) &= |Pr[\mathsf{Exp}(1) = 1] - Pr[\mathsf{Exp}(0) = 1]| \\
&= |Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] + Pr[\mathsf{Exp}(1) = 1|\divideontimes^c]Pr[\divideontimes^c] \\
&\quad - Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0) = 1|\divideontimes^c]Pr[\divideontimes^c]| \\
&\leq |Pr[\mathsf{Exp}(1) = 1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0) = 1|\divideontimes]Pr[\divideontimes]| \\
&\quad + |Pr[\mathsf{Exp}(1) = 1|\divideontimes^c]Pr[\divideontimes^c] - Pr[\mathsf{Exp}(0) = 1|\divideontimes^c]Pr[\divideontimes^c]|
\end{aligned}$$

$$
\begin{aligned}
&\leq |Pr[\mathsf{Exp}(1)=1|\divideontimes]Pr[\divideontimes] - Pr[\mathsf{Exp}(0)=1|\divideontimes]Pr[\divideontimes]| + Pr[\divideontimes^c] \\
&= |Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes] - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes]| + Pr[\divideontimes^c] \\
&= |Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes|\triangle]Pr[\triangle] + Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes|\triangle^c]Pr[\triangle^c] \\
&\quad - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes|\triangle^c]Pr[\triangle^c]| + Pr[\divideontimes^c] \\
&\leq |Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes|\triangle]Pr[\triangle]| \\
&\quad + |Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes|\triangle^c]Pr[\triangle^c] - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes|\triangle^c]Pr[\triangle^c]| + Pr[\divideontimes^c] \\
&\leq |Pr[\mathsf{Exp}(1)=1 \wedge \divideontimes|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1 \wedge \divideontimes|\triangle]Pr[\triangle]| + Pr[\triangle^c] + Pr[\divideontimes^c] \\
&\leq |Pr[\mathsf{Exp}(1)=1] - Pr[\mathsf{Exp}(0)=1]| + 2 \cdot Pr[\triangle^c] + 2 \cdot Pr[\divideontimes^c] \\
&= \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{A}}(q) + 2 \cdot Pr[\triangle^c] + 2 \cdot Pr[\divideontimes^c] \cdots (3) \\
\leq &\mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{A}}(q) + 2 \cdot (6 \cdot q \cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) + (3 \cdot q + 1) \cdot Pr[\divideontimes^c]) + 2 \cdot Pr[\divideontimes^c] \cdots (4)
\end{aligned}
$$

■

Claim ((3)→(4)) : $Pr[\triangle^c] \leq 6 \cdot q \cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) + (3 \cdot q + 1) \cdot Pr[\divideontimes^c]$.

To prove this Claim, we construct an adversary $\mathcal{C}_\mathcal{A}$ making ACPCA for $\mathrm{E}^*=\{e^{*i}\}_{i\in\mathbb{N}}$ in [E-12] and [E-13] as follows.

► Adversary $\mathcal{C}_\mathcal{A}$

$\mathcal{C}_\mathcal{A}^{E_a^*(\cdot),E_a^{*-1}}(find)$: Choose $t \in \{0,1,2,\cdots,q\}$ and $w \in \{1,2,3\}$ randomly. Then run $\mathcal{A}^{E_a(\cdot),E_a{}^{-1}}(find)$. $\mathcal{C}_\mathcal{A}$ simulates $\mathcal{A}$'s oracles $E_a(\cdot)$ and $E_a^{-1}(\cdot)$ with $\mathcal{C}_\mathcal{A}$'s oracles $E_a^*(\cdot)$ and $E_a^{*-1}(\cdot)$. For each $\mathcal{A}$'s query (x_i, r_i), $\mathcal{C}_\mathcal{A}$ gives this query to oracles $E_a^*(\cdot)$ and $E_a^{*-1}(\cdot)$. If $r_i = 1$, then $\mathcal{C}_\mathcal{A}$ obtains $E_a^*(x_i)$ from the oracle. If $r_i = -1$, then $\mathcal{C}_\mathcal{A}$ obtains $E_a^{*-1}(x_i)$ from the oracle. Repeat this process for $1 \leq i \leq t-1$. And then, in case $w = 1$, $\mathcal{C}_\mathcal{A}$ generates $x^0, x^1 \notin \{x_i | r_i = 1, 1 \leq i \leq t-1\} \cup \{E_a^{*-1}(x_j) | r_j = -1, 1 \leq j \leq t-1\}$ and gives $(x^0, x^1, 1)$ to $\mathcal{C}_\mathcal{A}$'s oracle. $\mathcal{C}_\mathcal{A}$ obtains $y = E_a^*(x^b)$ from the oracle. In case $w = 2$, $\mathcal{C}_\mathcal{A}$ runs $\mathcal{A}^{E_a(\cdot),E_a{}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_\mathcal{A}$ gives $(x_t, 1)$ to the oracle and obtains $T = E_a^*(x_t)$ from the oracle. Then $\mathcal{C}_\mathcal{A}$ generates $x^0, x^1 \notin \{x_i | r_i = 1, 1 \leq i \leq t-1\} \cup \{E_a^{*-1}(x_j) | r_j = -1, 1 \leq j \leq t-1\} \cup \{x_t\}$ and gives $(x^0, x^1, 1)$ to $\mathcal{C}_\mathcal{A}$'s oracle. $\mathcal{C}_\mathcal{A}$ obtains $y = E_a^*(x^b)$ from the oracle. In case $w = 3$, $\mathcal{C}_\mathcal{A}$ runs $\mathcal{A}^{E_a(\cdot),E_a{}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_\mathcal{A}$ gives $(x_t, -1)$ to the oracle and obtains $T = E_a^{*-1}(x_t)$ from the oracle. Then $\mathcal{C}_\mathcal{A}$ generates $x^0, x^1 \notin \{E_a^*(x_i) | r_i = 1, 1 \leq i \leq t-1\} \cup \{x_j | r_j = -1, 1 \leq j \leq t-1\} \cup \{E_a^*(x_t)\}$ and gives $(x^0, x^1, -1)$ to $\mathcal{C}_\mathcal{A}$'s oracle. $\mathcal{C}_\mathcal{A}$ obtains $y = E_a^{*-1}(x^b)$ from the oracle.

$\mathcal{C}_\mathcal{A}^{E_a^*(\cdot),E_a^{*-1}}(guess)$: In case $w = 1$, $\mathcal{C}_\mathcal{A}$ runs $\mathcal{A}^{E_a(\cdot),E_a{}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_\mathcal{A}$ assumes $x_t = a$ and $\mathcal{C}_\mathcal{A}$ calculates $E_{x_t}^*(x^0)$ and $E_{x_t}^*(x^1)$. If $E_{x_t}^*(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly. In case $w = 2$, $\mathcal{C}_\mathcal{A}$ assumes $x_t = E_a^{*-1}(a)$ and calculates $E_T^*(x^0)$ and $E_T^*(x^1)$. If $E_T^*(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly. In case $w = 3$, $\mathcal{C}_\mathcal{A}$ assumes $x_t = E_a^*(a)$ and calculates $E_T^{*-1}(x^0)$ and $E_T^{*-1}(x^1)$. If $E_T^{*-1}(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly.

Then, we check the following inequalities easily.

$$
\begin{aligned}
\tfrac{1}{3\cdot q} Pr[\triangle^c|\divideontimes] + \tfrac{1}{2}(1 - \tfrac{1}{3\cdot q} Pr[\triangle^c|\divideontimes]) &\leq Pr[\mathsf{Exp}^{FtG-ACPCA-b}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) = b|\divideontimes], \\
\tfrac{1}{2}(1 - \tfrac{1}{3\cdot q} Pr[\triangle^c|\divideontimes]) &\geq Pr[\mathsf{Exp}^{FtG-ACPCA-b}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) = b \oplus 1|\divideontimes]
\end{aligned}
$$

Therefore,(Here, $\mathsf{Exp}^*(b) = c$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) = c$.)

$$\begin{aligned}\frac{1}{3\cdot q}Pr[\triangle^c|\divideontimes] &\le |Pr[\mathsf{Exp}^*(b)=b|\divideontimes] - Pr[\mathsf{Exp}^*(b)=b\oplus 1|\divideontimes]| \\ &\le \frac{1}{Pr[\divideontimes]}\{|Pr[\mathsf{Exp}^*(b)=b] - Pr[\mathsf{Exp}^*(b)=b\oplus 1]| + Pr[\divideontimes^c]\} \\ &= \frac{2}{Pr[\divideontimes]}|Pr[\mathsf{Exp}^*(1)=1] - Pr[\mathsf{Exp}^*(0)=1]| + \frac{Pr[\divideontimes^c]}{Pr[\divideontimes]} \\ &= \frac{2}{Pr[\divideontimes]}\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) + \frac{Pr[\divideontimes^c]}{Pr[\divideontimes]} \cdots (5)\end{aligned}$$

And $\frac{1}{3\cdot q}Pr[\triangle^c|\divideontimes] \ge \frac{1}{3\cdot q\cdot Pr[\divideontimes]}(Pr[\triangle^c] - Pr[\divideontimes^c]) \cdots (6)$.

By (5) and (6), $\frac{1}{3\cdot q\cdot Pr[\divideontimes]}(Pr[\triangle^c] - Pr[\divideontimes^c]) \le \frac{2}{Pr[\divideontimes]}\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) + \frac{Pr[\divideontimes^c]}{Pr[\divideontimes]}$.

Therefore, $Pr[\triangle^c] \le 6\cdot q\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q) + (3\cdot q+1)\cdot Pr[\divideontimes^c]$. ■

3. Case [E-14]

For [E-14],

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \le (12\cdot q+1)\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1}}(q)$$

Proof. In case [E-14], we only consider block cipher E^* such that $\{x|(x = a)\vee(x = a\oplus h_0)\}\cap\{x|(x = E_a^{*-1}(a))\vee(x = E_a^{*-1}(a\oplus h_0))\} = \phi$. $\triangle$ means the event that there is no query x of an adversary $\mathcal{A}$ among maximum q queries such that $x\in\{x|(x=a)\vee(x=a\oplus h_0)\}\cup\{x|(x=E_a^{*-1}(a))\vee(x=E_a^{*-1}(a\oplus h_0))\}\cup\{E_a^*(E_a^*(x))|(x=E_a^{*-1}(a))\vee(x=E_a^{*-1}(a\oplus h_0))\}$. $\mathsf{Exp}(b)=1$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E,E^{-1},\mathcal{A}}(q)=1$ in the following inequalities.

$$\begin{aligned}\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) &= |Pr[\mathsf{Exp}(1)=1] - Pr[\mathsf{Exp}(0)=1]| \\ &= |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] + Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] \\ &\quad - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\ &\le |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle]| \\ &\quad + |Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\ &\le |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle]| + Pr[\triangle^c] \\ &\le |Pr[\mathsf{Exp}(1)=1] - Pr[\mathsf{Exp}(0)=1]| + 2\cdot Pr[\triangle^c] \\ &= \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{A}}(q) + 2\cdot Pr[\triangle^c]\cdots(7) \\ &\le \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{A}}(q) + 2\cdot(6\cdot q\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q))\cdots(8)\end{aligned}$$ ■

Claim ((7)→(8)) : $Pr[\triangle^c] \le 6\cdot q\cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*,E^{*-1},\mathcal{C}_\mathcal{A}}(q)$.

To prove this Claim, we construct an adversary $\mathcal{C}_\mathcal{A}$ making ACPCA for E*=$\{e^{*i}\}_{i\in\mathbb{N}}$ in [E-14] as follows.

► Adversary $\mathcal{C}_\mathcal{A}$

$\mathcal{C}_\mathcal{A}^{E_a^*(\cdot),E_a^{*-1}}(find)$: Choose $t\in\{0,1,2,\cdots,q\}$ and $w\in\{1,2,3\}$ randomly. Then run $\mathcal{A}^{E_a(\cdot),E_a^{-1}}(find)$. $\mathcal{C}_\mathcal{A}$ simulates $\mathcal{A}$'s oracles $E_a(\cdot)$ and $E_a^{-1}(\cdot)$ with $\mathcal{C}_\mathcal{A}$'s oracles $E_a^*(\cdot)$ and $E_a^{*-1}(\cdot)$. For each $\mathcal{A}$'s query (x_i,r_i), $\mathcal{C}_\mathcal{A}$ gives this query to oracles $E_a^*(\cdot)$ and $E_a^{*-1}(\cdot)$. If $r_i=1$, then $\mathcal{C}_\mathcal{A}$ obtains $E_a^*(x_i)$ from the oracle. If $r_i=-1$,

then $\mathcal{C}_{\mathcal{A}}$ obtains ${E_a^*}^{-1}(x_i)$ from the oracle. Repeat this process for $1 \leq i \leq t-1$. And then, in case $w = 1$, $\mathcal{C}_{\mathcal{A}}$ generates $x^0, x^1 \notin \{x_i | r_i = 1, 1 \leq i \leq t-1\} \cup \{{E_a^*}^{-1}(x_j) | r_j = -1, 1 \leq j \leq t-1\}$ and gives $(x^0, x^1, 1)$ to $\mathcal{C}_{\mathcal{A}}$'s oracle. $\mathcal{C}_{\mathcal{A}}$ obtains $y = E_a^*(x^b)$ from the oracle. In case $w = 2$, $\mathcal{C}_{\mathcal{A}}$ runs $\mathcal{A}^{E_a(\cdot), {E_a}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_{\mathcal{A}}$ gives $(x_t, 1)$ to the oracle and obtains $T = E_a^*(x_t)$ from the oracle. Then $\mathcal{C}_{\mathcal{A}}$ generates $x^0, x^1 \notin \{x_i | r_i = 1, 1 \leq i \leq t-1\} \cup \{{E_a^*}^{-1}(x_j) | r_j = -1, 1 \leq j \leq t-1\} \cup \{x_t\}$ and gives $(x^0, x^1, 1)$ to $\mathcal{C}_{\mathcal{A}}$'s oracle. $\mathcal{C}_{\mathcal{A}}$ obtains $y = E_a^*(x^b)$ from the oracle. In case $w = 3$, $\mathcal{C}_{\mathcal{A}}$ runs $\mathcal{A}^{E_a(\cdot), {E_a}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_{\mathcal{A}}$ gives $(x_t, -1)$ to the oracle and obtains $T = {E_a^*}^{-1}(x_t)$ from the oracle. Then $\mathcal{C}_{\mathcal{A}}$ generates $x^0, x^1 \notin \{E_a^*(x_i) | r_i = 1, 1 \leq i \leq t-1\} \cup \{x_j | r_j = -1, 1 \leq j \leq t-1\} \cup \{E_a^*(x_t)\}$ and gives $(x^0, x^1, -1)$ to $\mathcal{C}_{\mathcal{A}}$'s oracle. $\mathcal{C}_{\mathcal{A}}$ obtains $y = {E_a^*}^{-1}(x^b)$ from the oracle.

$\mathcal{C}_{\mathcal{A}}^{E_a^*(\cdot), {E_a^*}^{-1}}(guess)$: In case $w = 1$, $\mathcal{C}_{\mathcal{A}}$ runs $\mathcal{A}^{E_a(\cdot), {E_a}^{-1}}(find)$ one more and obtains (x_t, r_t) from $\mathcal{A}$. $\mathcal{C}_{\mathcal{A}}$ assumes $x_t = a$ or $x_t = a \oplus h_0$ and $\mathcal{C}_{\mathcal{A}}$ calculates $E_{x_t}^*(x^0)$, $E_{x_t}^*(x^1)$, $E_{x_t \oplus h_0}^*(x^0)$ and $E_{x_t \oplus h_0}^*(x^1)$. If $E_{x_t}^*(x^s) = y$ or $E_{x_t \oplus h_0}^*(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly. In case $w = 2$, $\mathcal{C}_{\mathcal{A}}$ assumes $x_t = {E_a^*}^{-1}(a)$ or $x_t = {E_a^*}^{-1}(a \oplus h_0)$ and calculates $E_T^*(x^0)$, $E_T^*(x^1)$, $E_{T \oplus h_0}^*(x^0)$ and $E_{T \oplus h_0}^*(x^1)$. If $E_T^*(x^s) = y$ or $E_{T \oplus h_0}^*(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly. In case $w = 3$, $\mathcal{C}_{\mathcal{A}}$ assumes $x_t = E_a^*(a)$ or $x_t = E_a^*(a \oplus h_0)$ and calculates ${E_T^*}^{-1}(x^0)$, ${E_T^*}^{-1}(x^1)$, ${E_{T \oplus h_0}^*}^{-1}(x^0)$ and ${E_{T \oplus h_0}^*}^{-1}(x^1)$. If ${E_T^*}^{-1}(x^s) = y$ or ${E_{T \oplus h_0}^*}^{-1}(x^s) = y$, then output $b' = s$, otherwise output $b' = 0$ *or* 1 randomly.

Then, we check the following inequalities easily.

$$\frac{1}{3 \cdot q} Pr[\triangle^c] + \frac{1}{2}\left(1 - \frac{1}{3 \cdot q} Pr[\triangle^c]\right) \leq Pr[\mathsf{Exp}^{FtG-ACPCA-b}_{E^*, {E^*}^{-1}, \mathcal{C}_{\mathcal{A}}}(q) = b],$$
$$\frac{1}{2}\left(1 - \frac{1}{3 \cdot q} Pr[\triangle^c]\right) \geq Pr[\mathsf{Exp}^{FtG-ACPCA-b}_{E^*, {E^*}^{-1}, \mathcal{C}_{\mathcal{A}}}(q) = b \oplus 1]$$

Therefore,(Here, $\mathsf{Exp}^*(b) = c$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E^*, {E^*}^{-1}, \mathcal{C}_{\mathcal{A}}}(q) = c$.)

$$\begin{aligned} \frac{1}{3 \cdot q} Pr[\triangle^c] &\leq |Pr[\mathsf{Exp}^*(b) = b] - Pr[\mathsf{Exp}^*(b) = b \oplus 1]| \\ &= 2 \cdot |Pr[\mathsf{Exp}^*(1) = 1] - Pr[\mathsf{Exp}^*(0) = 1]| \\ &= 2 \cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*, {E^*}^{-1}, \mathcal{C}_{\mathcal{A}}}(q) \end{aligned}$$

Therefore, $Pr[\triangle^c] \leq 6 \cdot q \cdot \mathbf{Adv}^{FtG-ACPCA}_{E^*, {E^*}^{-1}, \mathcal{C}_{\mathcal{A}}}(q)$. ■

4. Case [E-15]

For [E-15],

$$\mathbf{Adv}^{FtG-ACPCA}_{E, E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E, E^{-1}}(q) + 2^{1-n}$$

Proof. It is easy to check. ■

5. Case [E-17]

For [E-17] (the input and output length of E^* is n. The length of key of E^* is $n-2$),

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q)$$

Proof. $\triangle$ means the event $a[high\ 2th\ bit] = 1$ and $\triangle^c$ means the event $a[high\ 2th\ bit] = 0$. $\mathsf{Exp}(b) = 1$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E,E^{-1},\mathcal{A}}(q) = 1$ in the following inequalities.

$$\begin{aligned}
\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) &= |Pr[\mathsf{Exp}(1)=1] - Pr[\mathsf{Exp}(0)=1]| \\
&= |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] + Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] \\
&\quad - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\
&\leq |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle]| \\
&\quad + |Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\
&\leq \tfrac{1}{2}|Pr[\mathsf{Exp}(1)=1|\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]| \\
&\quad + \tfrac{1}{2}|Pr[\mathsf{Exp}(1)=1|\triangle^c] - Pr[\mathsf{Exp}(0)=1|\triangle^c]|
\end{aligned}$$

Therefore,

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \tfrac{1}{2}\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) + \tfrac{1}{2}\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) = \mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q)$$ ■

6. Case [E-18]

For [E-18] (the input and output length of E^* is n. The length of key of E^* is $n-1$),

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q)$$

Proof. $\triangle$ means the event $a[high\ 1th\ bit] = 1$ and $\triangle^c$ means the event $a[high\ 1th\ bit] = 0$. $\mathsf{Exp}(b) = 1$ means $\mathsf{Exp}^{FtG-ACPCA-b}_{E,E^{-1},\mathcal{A}}(q) = 1$ in the following inequalities.

$$\begin{aligned}
\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1},\mathcal{A}}(q) &= |Pr[\mathsf{Exp}(1)=1] - Pr[\mathsf{Exp}(0)=1]| \\
&= |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] + Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] \\
&\quad - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\
&\leq |Pr[\mathsf{Exp}(1)=1|\triangle]Pr[\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]Pr[\triangle]| \\
&\quad + |Pr[\mathsf{Exp}(1)=1|\triangle^c]Pr[\triangle^c] - Pr[\mathsf{Exp}(0)=1|\triangle^c]Pr[\triangle^c]| \\
&\leq \tfrac{1}{2}|Pr[\mathsf{Exp}(1)=1|\triangle] - Pr[\mathsf{Exp}(0)=1|\triangle]| \\
&\quad + \tfrac{1}{2}|Pr[\mathsf{Exp}(1)=1|\triangle^c] - Pr[\mathsf{Exp}(0)=1|\triangle^c]|
\end{aligned}$$

Therefore,

$$\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) \leq \tfrac{1}{2}\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) + \tfrac{1}{2}\mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q) = \mathbf{Adv}^{FtG-ACPCA}_{E,E^{-1}}(q)$$ ■

The Security and Performance of the Galois/Counter Mode (GCM) of Operation

David A. McGrew[†] and John Viega[‡]

[†] Cisco Systems, Inc.,
mcgrew@cisco.com
[‡] Secure Software
viega@securesoftware.com

Abstract. The recently introduced Galois/Counter Mode (GCM) of operation for block ciphers provides both encryption and message authentication, using universal hashing based on multiplication in a binary finite field. We analyze its security and performance, and show that it is the most efficient mode of operation for high speed packet networks, by using a realistic model of a network crypto module and empirical data from studies of Internet traffic in conjunction with software experiments and hardware designs. GCM has several useful features: it can accept IVs of arbitrary length, can act as a stand-alone message authentication code (MAC), and can be used as an incremental MAC. We show that GCM is secure in the standard model of concrete security, even when these features are used. We also consider several of its important system-security aspects.

1 Introduction

The Galois/Counter Mode (GCM) of operation for block ciphers was designed to meet the need for an authenticated encryption mode that can efficiently achieve speeds of 10 gigabits per second and higher in hardware, can perform well in software, and is free of intellectual property restrictions. It was recently submitted to several standards venues, including the NIST Modes of Operation process [18], IEEE 802.1AE Link Security [21], where it is the mandatory-to-implement cryptoalgorithm in the current draft standard, and IPsec [24]. In the following, we consider its performance and security.

The counter mode of operation (CTR) has become the mode of choice for high speed applications, because it can be efficiently pipelined in hardware implementations. However, it provides no message authentication. GCM incorporates CTR and builds on it by adding a message authentication code (MAC) based on universal hashing [25, 16]. It uses polynomial hashing in the finite field $GF(2^w)$, the core operation of which is multiplication by a fixed field element. The binary field multiplication can be implemented easily in hardware, and can be made surprisingly efficient in software via table-driven methods. Additionally, GCM can be used as a stand-alone MAC, and can be used as an incremental MAC [3].

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 343–355, 2004.

This paper is structured as follows. In Section 1.1 we review existing work on authenticated encryption with associated data (AEAD) methods [19]. In Section 2 we briefly review the GCM definition. In Section 3 we analyze and describe its performance in hardware are software, and compare it to other AEAD modes of operation. In Section 4, we review our analysis of GCM in the concrete model; this paper is an extended abstract, and proofs of our results are provided in the full version. In Section 5 we consider several important system-security aspects.

1.1 Overview of Authenticated Encryption Modes

Recently, many authenticated encryption modes have been proposed, because of the efficiency and usability benefits of the combined approach. The first such mode was Jutla's IAPM (Integrity-Aware Parallelizable) mode [13]. The better known OCB (Offset Code Book) mode [20] is a refinement of IAPM. Both of these modes are parallelizable, making them suitable for high-speed hardware implementations (though they cannot take complete advantage of pipelining; see Section 3). Independently, Gligor and Donescu proposed several authenticated encryption modes [11]. All of the above modes are covered by patents, which has motivated some other work in this space. CCM [26] uses a single key and combines CTR mode with CBC-MAC to produce an authenticated encryption scheme. However, CCM is not suited to high-speed implementations, because CBC-MAC is neither pipelinable nor parallelizable. EAX [8] is a patent-free mode similar to CCM, combining CTR with the OMAC [12] variant of CBC-MAC. OMAC cannot be pipelined or parallelized, so neither can EAX. However, EAX solves some minor issues unique to CCM: it is not *on-line*, meaning that the message length must be known before one can start processing the message, and there are cases in which it does not preserve word alignment. CWC mode [15] is both patent-free and fully parallelizable; it combines CTR with a MAC based on a universal hash function over $GF(2^{127}-1)$. Due to its use of an integer multiply operation, CWC is relatively expensive to implement in hardware.

One useful feature of many authenticated encryption schemes is the ability to authenticate associated data that is not part of the message, such as packet headers. IAPM and OCB are the only two modes we have discussed that have no facilities for this. Another interesting feature, introduced by EAX, is the ability to accept arbitrary-length IVs (most modes use IVs no longer than the cipher block width). This facility increases the usability of the mode, but has the disadvantage of requiring additional processing - particularly in hardware, where a pipeline stall caused by IV processing can significantly impact throughput. GCM supports arbitrary sized IVs, but is optimized for the 12-byte case. As with most modes, GCM uses a single key, supports additional authenticated data, preserves data alignment in all cases, and is on-line.

GCM's design draws from several sources. It uses CTR for encryption, and uses a polynomial hash, like CWC, but with a relatively inexpensive binary field. Its architecture follows that of the Universal Security Transform [17], which enables it to be efficiently pipelined.

2 GCM Definition

We briefly review the definition of GCM, closely following its specification [18], but considering a block cipher with a width of $w \geq 64$ bits, instead of focusing on the 128-bit wide Advanced Encryption Standard (AES) [23]. We assume that w is even. The two main functions that GCM uses are block cipher encryption and multiplication over the field $GF(2^w)$; it defines a particular field, but its details are irrelevant to our analysis. The block cipher encryption of the value $X \in \{0,1\}^w$ with the key K is denoted as $E(K,X)$. The multiplication of two elements $X, Y \in GF(2^w)$ is denoted as $X \cdot Y$, and the addition of X and Y is denoted as $X \oplus Y$. The function $\mathrm{len}(S)$ takes a bit string S with a length between zero and $2^{w/2}-1$, inclusive, and returns a $w/2$-bit string containing the nonnegative integer describing the number of bits in its argument, with the least significant bit on the right. The expression 0^l denotes a string of l zero bits, and $A\|B$ denotes the concatenation of two bit strings A and B. The function $\mathrm{MSB}_t(S)$ takes a bit string S and returns the bit string containing only the leftmost t bits of S, and the symbol $\{\}$ denotes the bit string with zero length.

The authenticated encryption operation takes as inputs a secret key K, initialization vector IV, a plaintext P, and additional authenticated data A, and gives as its outputs a ciphertext C and an authentication tag T. These values are bit strings with lengths given as follows:

$$\begin{aligned} 0 \leq \mathrm{len}(P) &\leq (2^{32}-2)w \\ 0 \leq \mathrm{len}(A) &\leq 2^{w/2} \\ 0 < \mathrm{len}(IV) &\leq 2^{w/2} \\ \mathrm{len}(C) &= \mathrm{len}(P) \\ \mathrm{len}(T) &= t \leq w, \end{aligned} \tag{1}$$

where the parameter t is fixed for each instance of the key. The secret key has a length appropriate to the block cipher, and is only used as an input to that cipher. For each fixed value of K, each value of the IV must be distinct, but those values need not have equal lengths. The authenticated decryption operation has five inputs: K, IV, C, A, and T, as defined above. It has only one output, either the plaintext value P or the special symbol **FAIL** that indicates that its inputs are not authentic.

During the encryption and decryption processes, the bit strings P, C, and A are broken up into w-bit blocks. We let n and u denote the unique pair of positive integers such that the total number of bits in the plaintext is $(n-1)w+u$, where $1 \leq u \leq w$, when $\mathrm{len}(P) > 0$; otherwise $n = u = 0$. The plaintext consists of a sequence of n bit strings, in which the bit length of the last bit string is u, and the bit length of the other bit strings is w. The sequence is denoted $P_1, P_2, \ldots, P_{n-1}, P_n^*$, and the bit strings are called data blocks, although the last bit string, P_n^* , may not be a complete block. Similarly, the ciphertext is denoted as $C_1, C_2, \ldots, C_{n-1}, C_n^*$, where the number of bits in the final block C_n^* is u. The additional authenticated data A is denoted as $A_1, A_2, \ldots, A_{m-1}, A_m^*$,

where the last bit string A_m^* may be a partial block of length v, and m and v denote the unique pair of positive integers such that the total number of bits in A is $(m-1)w+v$ and $1 \leq v \leq w$, when $\text{len}(A) > 0$; otherwise $m = v = 0$. The authenticated encryption operation is defined by the following equations:

$$\begin{aligned} H &= E(K, 0^w) \\ Y_0 &= \begin{cases} IV \| 0^{31} 1 & \text{if } \text{len}(IV) = w - 32 \\ \text{GHASH}(H, \{\}, IV) & \text{otherwise.} \end{cases} \\ Y_i &= \text{incr}(Y_{i-1}) \text{ for } i = 1, \ldots, n \\ C_i &= P_i \oplus E(K, Y_i) \text{ for } i = 1, \ldots, n-1 \\ C_n^* &= P_n^* \oplus \text{MSB}_u(E(K, Y_n)) \\ T &= \text{MSB}_t(\text{GHASH}(H, A, C) \oplus E(K, Y_0)) \end{aligned} \tag{2}$$

Successive counter values are generated using the function incr(), which treats the rightmost 32 bits of its argument as a nonnegative integer with the least significant bit on the right, and increments this value modulo 2^{32}. More formally, the value of $\text{incr}(F \| I)$ is $F \| (I + 1 \bmod 2^{32})$. The function GHASH is defined by $\text{GHASH}(H, A, C) = X_{m+n+1}$, where the inputs A and C are formatted as described above, and the variables X_i for $i = 0, \ldots, m+n+1$ are defined as

$$X_i = \begin{cases} 0 & \text{for } i = 0 \\ (X_{i-1} \oplus A_i) \cdot H & \text{for } i = 1, \ldots, m-1 \\ (X_{m-1} \oplus (A_m^* \| 0^{w-v})) \cdot H & \text{for } i = m \\ (X_{i-1} \oplus C_{i-m}) \cdot H & \text{for } i = m+1, \ldots, m+n-1 \\ (X_{m+n-1} \oplus (C_n^* \| 0^{w-u})) \cdot H & \text{for } i = m+n \\ (X_{m+n} \oplus (\text{len}(A) \| \text{len}(C))) \cdot H & \text{for } i = m+n+1. \end{cases} \tag{3}$$

3 Performance

We considered the performance of various modes of operation of the AES-128 block cipher in both hardware and software. We use a simple model of a network crypto module in order to analyze the performance of different AEAD modes under realistic conditions. The module consists of a device that accepts a continuous stream unprotected data packets on one interface and then outputs the stream of encrypted and authenticated packets out another interface. We assume that the key is present in the module, and that the mode and data encapsulation are fixed, in order to focus on the data processing performance. We assume that the module contains a clock which runs at a fixed rate. In general, the number of clock cycles $C(s)$ required to process a packet with s bytes varies as a function of s. We assume that the packet sizes are distributed probabilistically, where the probability of having size s is $\mathbf{P}[S = s]$. The expected number of clocks per byte C of the module is $C = \sum_s C(s) f(s)$, where

$$f(s) = \frac{\mathbf{P}[S = s]}{\sum_r r\mathbf{P}[S = r]} \tag{4}$$

is the expected fraction of bytes that are carried in packets of size s. The function $f(s)$ is important because it can be empirically observed. Studies of Internet traffic reveal a predominance of small packets, with peaks in the distribution of packet sizes at 44, 552, 576, and 1500 bytes, and very few packets with larger sizes [9], reflecting the nature of the TCP/IP protocol family. About half of the data on the Internet is carried in packets of 576 bytes or less, and most of the remainder is carried in packets of about 1500 bytes. We defined the *Internet Performance Index* (IPI) as the expected number of bits processed per clock cycle when the packet distribution has the values $f(1500) = .6$, $f(576) = .2$, $f(552) = .15$, and $f(44) = .05$, using data from [9]. This index is a useful indicator of the performance of a crypto module that protects IP traffic using e.g. the Encapsulating Security Payload (ESP) [14] in tunnel mode, the protocol which underlies most Virtual Private Networks.

3.1 Hardware

A typical high-speed AES-128 implementation consists of a pipeline of ten units, each of which implements a single AES round. At each clock cycle, data moves from one unit to the next, and 128 bits enter the pipeline and the same number leave the pipeline as output. In the following, we describe and analyze the best GCM, CWC, and OCB implementations that use a single instance of this AES pipeline. We disregard the other modes, since they use cipher block chaining and thus cannot be implemented in this manner. We require that packets be processed sequentially, rather than concurrently, because the complexity and circuit cost of the concurrent approach makes sequential processing more desirable in practice. In our performance analysis we determine the value of $C(s)$ for each mode and tabulate the results, and also compare the circuit costs for the modes. Data from multiple packets may be in the module simultaneously. To account for this fact, we measure $C(s)$ by counting the number of cycles between the time when the last data from one protected packet leaves the module and when the initial data from the next protected packet leaves the module. Our hardware implementation model is not detailed, but it very effectively reveals the effects of pipeline stalls on performance; a *stall* occurs when a circuit is not generating outputs for some number of clock cycles.

GCM can easily take advantage of the AES pipeline (Figure 1, top), as long as a 96-bit IV is used (as is recommended for high-speed implementations). We use a finite-field multiplier over $GF(2^w)$ that executes in a single clock cycle. An important property of the mode is that the counter Y_0 that is used to encrypt the GHASH output can immediately follow the other counters through the AES pipeline, so that after the plaintext is encrypted, only one additional clock is needed to compute the authentication tag. Thus GCM can achieve $C(s) = \lceil s/16 \rceil + 1$ by having the data from each packet immediately follow that of the previous packet through the pipeline. CWC is similar but requires an additional AES encryption to process the authentication tag. This causes a

pipeline stall of 10 clock cycles during which the tag passes through the AES pipeline; thus for CWC, $C(s) = \lceil s/16 \rceil + 11$.

In OCB (Figure 1, bottom), the AES pipline is used in three distinct ways: to encrypt the IV, to encrypt the plaintext, and to compute the authentication tag. The pipeline stalls for ten clock cycles while the IV is being encrypted. After that computation, the stall continues for another ten clock cycles, until the plaintext that is being encrypted appears at the output of the pipeline as ciphertext. After all of the plaintext has been encrypted, the 'checksum' value is encrypted; this operation requires only a single clock cycle, because the data from the IV-encryption of the next packet can follow the data from the checksum-encryption through the pipeline. Thus OCB can achieve $C_{\mathrm{OCB}} = \lceil s/16 \rceil + 21$. In Table 1,

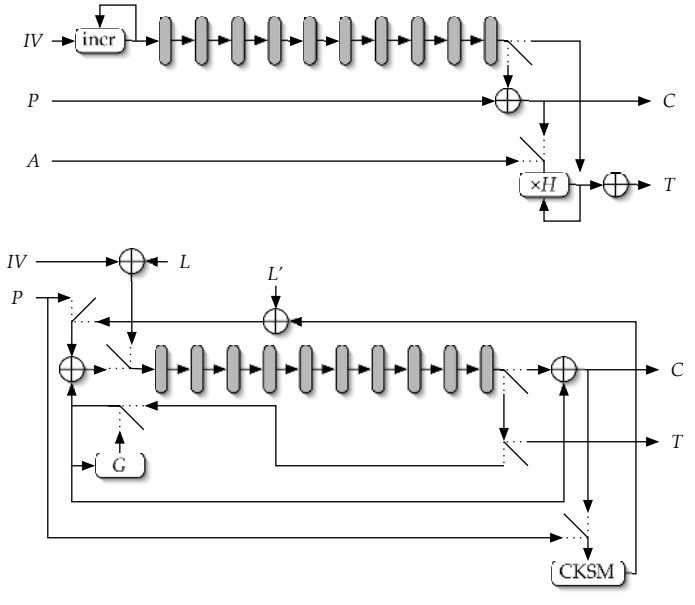

Fig. 1. Pipelined high-speed implementations of AES-128 GCM (top) and AES-128 OCB (bottom). During each clock cycle, 128 bits of data move across each arrow. Some details have been omitted for clarity

we compare the GCM, CWC, and OCB implementations described above. Various data sizes are included, along with the Internet Performance Index, and throughput is shown in bits per clock cycle. GCM excels the other modes in every category, especially at shorter lengths, because it keeps its pipeline full. In a crypto module that can process 128 bits per clock cycle, an ten-cycle pipeline stall has a considerable opportunity cost: 160 bytes could be encrypted during that time. GCM performance on the IPI is over twice that of CWC and over three times that of OCB. The circuit cost of GCM is higher than that of OCB because of its finite-field multiplier, but GCM is still the most economical mode

for high-speed operation. Even in the unlikely case that this multiplier required a circuit as large as the entire AES pipeline, a single GCM instance would have higher throughput on Internet data than three OCB implementations, while having less total circuit area. The cost of a single-clock $GF(2^{128})$-multiplier has been estimated at 30% of the cost of the AES-128 pipeline; a detailed analysis of this cost is beyond the scope of this paper. The circuit cost of CWC is significantly higher than that of GCM because it uses an integer multiplier rather than a binary-field multiplier.

Table 1. Hardware performance in bits per clock cycle, with three significant digits, for a variety of packet sizes and the Internet Performance Index (IPI)

Bytes	16	20	40	44	64	128	256	552	576	1024	1500	8192	IPI
GCM	64.0	71.1	91.4	93.9	102	114	120	124	124	126	127	128	77.7
CWC	10.7	13.1	23.7	25.6	34.1	53.9	75.9	97.0	98.0	109	115	125	35.3
OCB	5.82	7.19	13.6	14.8	20.5	35.3	55.4	79.6	80.8	96.4	105	123	22.8

3.2 Software

We tested software implementations of GCM, EAX, CCM, CWC, and OCB, each instantiated with the AES-128 cipher [23]. We also included CBC with HMAC-SHA1 to represent common current practice. We used the best available implementation of each mode, modified to use the fastest available AES implementation. All experiments took place on a 1Ghz Motorola G4 CPU using the GNU C compiler version 3.3 with full optimization. In this enviornment, AES-128 itself ran at 25 cycles per byte[1]. Qualitatively similar results were found on an Intel P4 [10], though CWC performed better on that CPU[2]. We tested GCM with both of the GHASH implementation strategies described in its specification, using 256 byte and 4Kb tables with Shoup's method [22] and 64Kb with the straightforward method. Table 2 shows our results. GCM has the best performance for the Internet Performance Index and on packets up to 576 bytes, while OCB has the best performance on larger packets. This result is easy to understand: OCB uses one more AES encryption per packet, while GCM does a $GF(2^w)$-multiply operation per block that OCB does not. The point at which their performance is equal reflects the number of multiplies that can be done in the time taken for a single AES encryption.

3.3 Other Applications

GCM can be used in an authentication-only mode, in which the data to be authenticated is included in A and the plaintext has zero length. In this mode

[1] Faster implementations have been reported for some CPUs, but are not publicly available. The table-driven GHASH algorithm, which uses the same basic operations as AES, may be able to benefit from similar implementation techniques.

[2] Gladman's Intel implementations used Bernstein's floating-point multiplication techniques [6], which provide significant advantages on some processors.

Table 2. Software performance in bits per kilocycle (or equivalently, megabits per second on a 1GHz processor) to three significant digits, on various packet sizes, and the Internet Performance Index (IPI), for various AES-128 modes of operation. GCM256, GCM4K, and GCM64K refer to GCM with 256, 4K, and 64K byte table sizes, respectively. The highest entry in each column is highlighted

Bytes	16	20	40	44	64	128	256	552	576	1024	1500	8192	IPI
GCM64K	**136**	**167**	**227**	**253**	**223**	**263**	**267**	**273**	**273**	266	266	258	**268**
GCM4K	116	140	190	207	192	213	229	237	233	239	247	240	240
GCM256	88.4	107	148	160	177	162	171	183	184	181	183	182	182
OCB	89.5	85.7	140	150	185	225	255	261	265	**273**	**275**	**282**	260
CWC	45.7	51.9	73.4	75.5	88.1	104	116	127	126	131	124	135	121
EAX	46.0	44.9	73.4	80.0	102	129	148	157	160	165	167	174	156
CCM	91.3	88.9	123	133	142	171	163	168	168	174	172	175	168
CBC-HMAC	6.3	8.0	15.2	16.6	23.4	39.0	64.5	96.0	97.0	117	129	156	88.6

(called GMAC), GCM has even more compelling advantages over most other modes because it avoids calling the block cipher once per block of data. (CWC is the only other mode with this property.) For instance, GMAC in our software test environment can process 1500-byte packets in 10.2 cycles per byte, whereas PMAC, a stand-alone MAC based on OCB [7], requires 27.6 cycles per byte.

The Secure Real-time Transport Protocol (SRTP) encrypts and authenticates real-time traffic, such as conversational voice, at the transport layer [1]. Typical plaintext sizes for this traffic include 20 bytes for the G.729 encoding, and 80 bytes for the G.721 encoding; GCM's performance on short packets makes it ideal for this application.

4 Security

The security of GCM stands on a single cryptographic conjecture: the block cipher E is assumed to be a secure pseudorandom permutation (PRP). To paraphrase Occam, we do not multiply conjectures beyond necessity. This requirement is met when E cannot be distinguished from a random permutation by an adversary that can choose its inputs and view its outputs. To formalize this idea, we use standard definitions from concrete security analysis, following [4]. The *permutation oracle* has the same interface as does the block cipher E with a fixed key. It takes as input a plaintext in $\{0,1\}^w$ and returns a ciphertext in $\{0,1\}^w$. We consider the experiment in which the adversary is given access to a permutation oracle and is challenged to determine whether it is the block cipher E with a randomly selected key (we denote this event as B_E), or a random permutation (which we denote as B_E^c). Each of these cases occurs with probability of 1/2. During the experiment, the adversary makes queries to the oracle and receives its responses. Afterwards, the adversary returns a bit that indicates its guess as to the content of the oracle. We denote as D the event that it guesses that B_E occurred, and denote as D^c the guess that B_E^c occured.

We define the *distinguishing advantage* A_E as the adversary's true positive probability less her false positive probability, that is,

$$A_E = \mathbf{P}[D \mid B_E] - \mathbf{P}[D \mid B_E^c]. \tag{5}$$

Here we use the conventional notation that $\mathbf{P}[\mathcal{X}]$ denotes the probability that the event $\mathcal{X}$ occurs, and $\mathbf{P}[\mathcal{X} \mid \mathcal{Y}] = \mathbf{P}[\mathcal{X} \cap \mathcal{Y}]/\mathbf{P}[\mathcal{Y}]$ denotes the probability that $\mathcal{X}$ occurs, given that the event $\mathcal{Y}$ has occured. We also use $\mathcal{X} \cap \mathcal{Y}$ to denote the event in which both events $\mathcal{X}$ and $\mathcal{Y}$ occur, and use $\mathcal{X}^c$ to denote the complement of $\mathcal{X}$, that is, the event that $\mathcal{X}$ does not occur. We make the simplifying assumption that $A_E > 0$, because an adversary that is consistently wrong can turn itself into one that is consistently right by just inverting its output. Thus the value A_E ranges between 0 and 1, inclusive.

Our model for the security of an AEAD system follows Rogaway [19]. The *authenticated encryption oracle* models the GCM authenticated encryption operation. It takes as input the bit strings IV, A, and P and returns the bit strings C and T, whose lengths obey the restrictions of Equations 1. The *authenticated decryption oracle* accepts inputs of the form (IV, A, C, T) and returns as its outputs either the special symbol **FAIL** or the plaintext P, where all of the bit strings are as defined above. We let the adversary choose the IVs, but assume that she is *nonce-respecting* and will not submit the same IV value to the same oracle multiple times (though she is free to submit a value to both oracles). We allow the adversary to interleave queries to these oracles. For our definition of confidentiality, we use the indistinguishability of ciphertext from random under a chosen plaintext attack and indistinguishability of plaintext from random under a chosen ciphertext attack. This strong definition has been shown to be equivalent to several other definitions [2]. Under these assumptions, GCM encryption is secure if an adversary presented with these oracles cannot tell if they contain GCM with a randomly selected key (we denote this event as B_{GCM}) or if C and T are a random function of the other inputs (which we denote as B_{GCM}^c). Each of these cases occurs with probability 1/2. Because GCM is not a generic composition of a cipher and a MAC, we cannot use the results of Bellare and Namprempre [5]. Most importantly, the use of the same secret value H for both hashing the IV and for computing the authentication tag provides the adversary a potential attack vector against confidentiality. For this reason, we need to give adversary access to the authenticated decryption oracle.

GCM uses E as a pseudorandom function (PRF). In our analysis, we make use of the well-known result on the use of a PRP as a PRF [4]. Our definition of PRF security considers the experiment in which we are given access to the *function oracle*, and are challenged to determine whether it contains a true random function or a PRF. That oracle has the same interface as does the permutation oracle; unlike that oracle, the function oracle may not be invertible. We use the convention that B_{PRF} denotes the PRF case and B_{PRF}^c denotes the random function case. The advantage of a PRF-distinguisher is given by $A_{\text{PRF}} = \mathbf{P}[D \mid B_{\text{PRF}}] - \mathbf{P}[D \mid B_{\text{PRF}}^c]$. The distinguishing advantage against a PRF is similar to that against a PRP, and has similar properties. The following Lemma bounds A_{PRF} in terms of A_E.

Lemma 1 (A PRP can be a Good PRF). *The advantage A_{PRF} of an adversary in distinguishing a w-bit PRP E from a random function is bounded by $A_{PRF} \leq A_E + q(q-1)2^{-w-1}$, where A_E is the adversary's advantage in distinguishing E from a random permutation, and the value q is the number of queries to the function oracle.*

Theorem 1 (GCM Encryption Is Secure). *If there is an adversary that can distinguish GCM encryption from a random function with advantage A_{GCM}, when the output of that function is limited to q queries to the authenticated encryption and decryption oracles, where the total number of plaintext bits processed is l_P and where $len(C) + len(A) \leq l$ and $len(IV) \leq l_{IV}$ for each query, then that adversary can distinguish E from a random permutation with advantage A_E, where*

$$A_E \geq A_{GCM} - (l_P/w + 2q)^2 2^{-w-1} - q((l_P/w + 2q)\lceil l_{IV}/w + 1\rceil 2^{1-w} + \lceil l/w + 1\rceil 2^{-t}). \quad (6)$$

This result is similar to that for counter mode, with a term that is quadratic in l_{P}. It also has a term that is linear in both l_{P} and ql_{IV}, which is due to the fact that collisions in the counter values are more likely when the lengths of the IVs that are hashed becomes greater. This term is dominant when $ql_{\mathrm{IV}} > l_{\mathrm{P}}$. The implication is that when long IVs are used, fewer queries should be made before a key is changed. However, in most cases l_{IV} will be no greater than l, and thus the accommodation of variable length IVs comes at negligible security cost.

The authentication tag size t affects the security of GCM encryption, but its effect is relatively weak. The term containing 2^{-t} in the bound on A_E does not dominate that value as long as t is greater than about $w - \lg(q\lceil l/w\rceil + \lceil l_{\mathrm{IV}}/w\rceil)$.

4.1 Authentication

We use the standard model for the security of a MAC in the presence of a chosen-message attack, in which an adversary is given access to a tag generation oracle and a message/tag verification oracle. The adversary can pass messages to the tag generation oracle and construct any message/tag pairs that it likes and send these to the verification oracle. Queries to the oracles can be interleaved by the adversary, if desired. The forgery advantage F_{GCM} is the probability that the adversary can get the verification oracle to accept a message/tag pair other than one generated by the tag generation oracle, after making q queries to the tag-generation oracle and the verification oracle.

Theorem 2 (GCM Authentication Is Secure). *An adversary with forgery advantage F_{GCM} against GCM, when q, l_P, l and l_{IV} are as defined in Theorem 1, has a distinguishing advantage A_E against the pseudorandom permutation E used in GCM of at least $F_{GCM} - (l_P/w + 2q)^2 2^{-w-1} - q((l_P/w + 2q + 1)\lceil l_{IV}/w + 1\rceil 2^{1-w} + \lceil l/w + 1\rceil 2^{-t})$.*

Like most authentication modes, the forgery advantage has a term that is quadratic in the amount of data that is authenticated. This term is dominant whenever many short messages are processed, as is typical for network crypto modules. When very long messages are processed, the term proportional to l will dominate. This term is characteristic of MACs that are based on universal hashing.

4.2 AES GCM Security

To tie our analysis to current practice, we apply it to the AES GCM specification for IPsec [24], for which $l_{\mathrm{IV}} = 96$ and $t = 96$. Any of the AES key lengths (of 128, 192, and 256 bits) can be used; for each variant, the block width $w = 128$. We use the typical Internet maximum packet size of 1500 bytes ($l \leq 12000$). The security of AES-N-GCM (for N=128, 192, or 256) is captured in the following corollary.

Corollary 1. *If there are no attacks against AES-N that can distinguish it from a random permutation with advantage greater than $A_{AES\text{-}N}$, and no more than q packets are processed, then*

- *there are no distinguishing attacks against AES-N-GCM that work with distinguishing advantage greater than $A_{AES\text{-}N} + q^2 2^{-116} - q2^{-89.4}$, and*
- *there are no forgery attacks against AES-N-GCM that work with forgery advantage greater than $A_{AES\text{-}N} + q^2 2^{-116} - q2^{-89.4} - q2^{-128}$.*

In these equations, the key size appears implicitly in the value of $A_{\mathrm{AES}\text{-}N}$. To provide a concrete example, these results show that, if AES is indistinguishable from a random permutation, and fewer than 2^{48} packets are protected, then the attacker's advantage is no more than 2^{-18}.

5 Other Security Aspects

We next consider system-security aspects. Having shown GCM secure when used properly, we consider what can go wrong. One often overlooked aspect of mode security is the consequence of IV misuse. It is well known that reusing a key/IV pair in CTR results in a loss of confidentiality for the messages that used the common IV value. Since GCM is built on top of CTR, it shares this property. However, the reuse of an IV in the GCM authenticated encryption operation (e.g. on the sender's side) causes even worse problems. It allows the attacker to solve for the underlying GHASH key H, making subsequent forgeries trivial and also enabling the attacker to choose IVs that will cause colliding counters. However, the reuse of an IV in the authenticated decryption operation (on the receiver's side) does not cause this problem. If an attacker convinces a receiver to decrypt multiple messages with the same IV, she still cannot exploit this situation to glean information about H efficiently. Fortunately, it is often comparatively easy for a sender to protect against IV reuse, for example, by using

a simple message counter as an IV. Additionally, GCM's ability to accept an arbitrary-length IVs makes it easier to ensure all IVs are unique, by including any possible distinguishing information, no matter how verbose. Interestingly, CWC avoids some of these issues by using the underlying block cipher to encrypt the output of its universal hash function. But this aspect of its design is responsible for causing the pipeline stalls that significantly degrade CWC's performance.

It is possible that $H = E(K, 0^w) = 0$, and in this case, $\mathrm{GHASH}(H, A, C) = 0^w$ for all values of A and C. If E behaves as a random permutation, then the expected number of keys for which $H = 0^w$ is the fraction 2^{-w} times the number of keys. This fact does not degrade the effectiveness of the message authentication; it is implicitly dealt with in the proof of security. When $H = 0^w$, the authentication tags will not be predictable; that case is no easier to detect than any other value of the key. However, that value causes all IVs to hash to the same value (if 96-bit IVs are not used). For this reason, some users may want to avoid using that key, e.g. by using the convention that H is set to a fixed value whenever the zero value is detected at key setup time. Of course, that key is so unlikely to arise in practice that its effect on the bounds in the security proofs are negligible, and it is equally reasonable not to bother to check for it.

References

1. M. Baugher , D. McGrew, M. Naslund, E. Carrara, K. Norrman. "The Secure Real-time Transport Protocol," *IETF RFC 3711*, March 2004.
2. M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. "A concrete security treatment of symmetric encryption," In *Proceedings of the 38th FOCS*, IEEE Computer Society Press, 1997.
3. M. Bellare, O. Goldreich, and S. Goldwasser, "Incremental cryptography: the case of hashing and signing", *CRYPTO '94*, LNCS, Springer-Verlag, Aug. 1994.
4. M. Bellare, J. Kilian, P. Rogaway, "The Security of the Cipher Block Chaining Message Authentication Code," *J. Comput. Syst. Sci.* 61(3). pg. 362-399 (2000).
5. M. Bellare and C. Namprempre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," *ASIACRYPT '00*, Springer-Verlag, LNCS, 2000.
6. D. Bernstein. *Floating-point arithmetic and message authentication*, Manuscript, 2000. Available online at `http://cr.yp.to/papers.html#hash127`.
7. J. Black and P. Rogaway, "A Block-Cipher Mode of Operation for Parallelizable Message Authentication," *EUROCRYPT '02*, LNCS, Springer-Verlag, 2002.
8. M. Bellare, P. Rogaway, and D. Wagner, "A conventional authenticated-encryption mode", *Submission to NIST Modes of Operation process*, 2003.
9. K. Claffy, G. Miller, K. Thompson, "The nature of the beast: recent traffic measurements from an Internet backbone," *INET '98*, ISOC, 1998.
10. B. Gladman, "AES and Combined Encryption/Authentication Modes," Web Page, http://fp.gladman.plus.com/AES/index.htm, February, 2004.
11. V. Gligor and P. Donescu, "Fast encryption and authentication: XCBC encryption and XECB authentication modes," *Proceedings of the Fast Software Encryption Workshop - FSE '01*. Springer-Verlag, 2001.

12. T. Iwata, K. Kurosawa, "OMAC: One-Key CBC MAC," *Submission to NIST Modes of Operation Process*, 2002.
13. C. Jutla, "Encryption modes with almost free message integrity," *Advanced in Cryptology - EUROCRYPT '01*, Springer-Verlag, 2001.
14. S. Kent, R. Atkinson, "IP Encapsulating Security Payload (ESP)," *IETF Request For Comments (RFC) 2406*, November 1998.
15. T. Kohno, J. Viega, and D. Whiting, "The CWC-AES Dual-use Mode," *Submission to NIST Modes of Operation Process*, 2003.
16. H. Krawczyk, "LFSR-based hashing and authentication," In Y. Desmedt, editor, *CRYPTO '94*, LNCS, Springer-Verlag, Aug. 1994.
17. D. McGrew. "The Universal Security Transform," *IETF Internet Draft*, Work in Progress. Oct. 2002.
18. D. McGrew and J. Viega. "The Galois/Counter Mode of Operation (GCM)," *Submission to NIST Modes of Operation Process*, January, 2004.
19. P. Rogaway. "Authenticated encryption with associated data," In *Proceedings of the 9th CCS*, Nov. 2002.
20. P. Rogaway, M. Bellare, J. Black, and T. Krovetz, "OCB: a block-cipher mode of operation for efficient authenticated encryption," *ACM CCS*, 2001.
21. A. Romanow, Ed. "Media Access Control (MAC) Security", *IEEE 802.1AE*, Draft Standard, July, 2004.
22. V. Shoup, "On Fast and Provably Secure Message Authentication Based on Universal Hashing," *CRYPTO '96*, LNCS, Springer-Verlag, 1996.
23. U.S. National Institute of Standards and Technology. The Advanced Encryption Standard. *Federal Information Processing Standard* (FIPS) 197, 2002.
24. J. Viega and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec ESP," *IETF Internet Draft*, Work in Progress, April, 2004.
25. M. Wegman and L. Carter, "New hash functions and their use in authentication and set equality," *Journal of Computer and System Sciences*, 22:265279, 1981.
26. D. Whiting, N. Ferguson, and R. Housley, "Counter with CBC-MAC (CCM)," *Submission to NIST Modes of Operation Process*, 2002.

Revisiting Fully Distributed Proxy Signature Schemes*

Javier Herranz and Germán Sáez

Dept. Matemàtica Aplicada IV, Universitat Politècnica de Catalunya
C. Jordi Girona, 1-3, Mòdul C3, Campus Nord, 08034-Barcelona, Spain
{jherranz, german}@ma4.upc.es

Abstract. In a proxy signature scheme, a potential signer delegates his capabilities to a proxy signer, who can sign documents on behalf of him. The recipient of the signature verifies both identities: that of the delegator and that of the proxy signer. There are many proposals of proxy signature schemes, but security of them has not been considered in a formal way until the appearance of [2, 8].

If the entities which take part in a proxy signature scheme are formed by sets of participants, then we refer to it as a fully distributed proxy signature scheme [4].

In this work, we extend the security definitions introduced in [2] to the scenario of fully distributed proxy signature schemes, and we propose a specific scheme which is secure in this new model.

1 Introduction

Digital signature schemes provide authenticity, integrity and non-repudiation to digital communications. Sometimes, however, a user must sign messages during a certain period of time in which he is not able to do it. For example, if this user is in holidays or has technical problems with its computer.

Proxy signature schemes were introduced in [9] and give a solution to this problem. An original user delegates his signing capabilities to a different user, the proxy signer. In this delegation, some aspects such as the dates of validity or the kind of messages that the proxy will be able to sign on behalf of the original signer should be stated. Later, the proxy signer can sign messages which conform to the delegation, on behalf of the original user. The recipient of the signature must verify at the same time the delegation of the original signer and the authenticity of the proxy signer.

A trivial solution to this problem is the following: the original signer uses his secret key to sign a delegation message (containing the terms of the delegations, his public key, the proxy signer's public key, etc.), and sends the message and the signature to the proxy signer. Later, when the proxy signer must sign a message

* This work was partially supported by Spanish *Ministerio de Ciencia y Tecnología* under project TIC 2003-00866.

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 356–370, 2004.

on behalf of the original one, he uses his secret key to compute a standard signature on this message. The final proxy signature includes the delegation message with his signature, the specific message and the signature computed by the proxy signer. Of course, the goal when designing more elaborated proxy signature schemes is to improve the efficiency of this trivial solution, for example by shortening the length of the final proxy signature.

If the participants of the system are not individual users, but distributed entities, then we must consider *fully distributed* proxy signature schemes (introduced in [4]). The original entity is formed by a set of members, and if an authorized subset of them cooperate, then they can delegate the signing power of the whole entity to the proxy entity. Later, if some authorized subset of members of the proxy entity cooperate, then they can compute a proxy signature of a message on behalf of the original entity. Such schemes can be useful, for example, when the participants in the system are important companies, or a central office of a bank and the branch offices, etc. In these cases, it is usually undesirable that the power to perform such important tasks (signing, or delegating rights) is held by a unique person or machine.

Almost all the proxy signature schemes (either individual or distributed) that have been proposed until now lack a formal proof of security. This fact has led to many attacks on some of these schemes. Furthermore, this lack of formalism is not in compliance with the current techniques of public key cryptography, where the security of the protocols is formally proved (this is known as *provable security*). That is, both the capabilities and the goals of an adversary who tries to attack the cryptographic scheme must be clearly stated. Then, the security of the scheme should be proved by showing that a successful attack against it could be used as a part of another attack which would solve a computationally hard problem (discrete logarithm, integer factorization, etc.).

The first step in order to formalize individual proxy signature schemes has been taken in [2]. There, a formal model of security for this kind of schemes is given, along with some schemes which can be proved secure according to this model. Their model is valid for schemes with one level of delegation. In order to support chains of several levels of delegation, another formalization of the security of proxy signatures has been given in [8].

In this work, we concentrate on distributed proxy schemes with one level of delegation; we extend the results in [2], by giving a formal model of security for fully distributed proxy signature schemes. Then, we explain a distributed version of one of the schemes which are proposed and proved secure in [2]. We prove that this new scheme is secure in the security model for fully distributed proxy signature schemes.

Organization of the Paper. In Section 2, we review some aspects of proxy signature schemes, including a specific scheme proposed in [2], as well as some distributed protocols that we will use later. In Section 3, we formally define what a fully distributed signature scheme is, and we give the natural security model for these schemes, derived from the model given in [2]. In Section 4, we

propose a new fully distributed proxy signature scheme (its security is proven in the Appendix). The work ends with some comments and conclusions in Section 5.

2 Preliminaries

The mathematical framework of the specific protocols that we are going to explain is the following. There are two large prime numbers p and q such that $q|p-1$. We consider an element $g \in \mathbb{Z}_p^*$ whose order is exactly q. We additionally need two hash functions H_1 and H_2 which map arbitrarily long strings of bits into $\mathbb{Z}_q$.

2.1 Proxy Signatures

Since its introduction by Mambo et al. [9], proxy signature schemes have been developed in many papers (for example [5, 7, 6]). Most of the proposed schemes are based on discrete-logarithm type signature schemes, such as Schnorr's [11]. In this signature scheme, each signer has a secret key $x \in \mathbb{Z}_q^*$ and the corresponding public key $y = g^x \bmod p$. To sign a message M, this signer chooses a random value $k \in \mathbb{Z}_q^*$ and then he computes the values $r = g^k \bmod p$ and $s = k + xH_1(M, r) \bmod q$. The signature of the message M is the pair (r, s), and its correctness can be verified by checking the equation $g^s = ry^{H_1(M,r)} \bmod p$. We use the notation $(r, s) = Sch\_Sig(M, sk, H_1)$ to refer to an execution of this signature scheme for message M, with secret key sk and hash function H_1.

Schnorr's signature scheme has been shown [10] to achieve the highest level of security for signature schemes, which is existential unforgeability under chosen message attacks. However, all the proposals of proxy signature schemes have lacked a formal security analysis. These schemes have been considered secure just until some attack against them has appeared (see [6, 14]).

The situation has changed since the appearance of a paper by Boldyreva et al. [2]. There, formal definitions on proxy signature schemes and their security are given; in the rest of this work, we follow the notation of [2].

Let $Sig = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ be a standard signature scheme. That is, $\mathcal{G}$ is the parameter-generator, which takes as input a security parameter and outputs some global parameters of the scheme (in our scenario, the prime numbers p and q, etc.). The key-generator $\mathcal{K}$ takes as input the global parameters and outputs a secret-public key pair (sk, pk). The signing algorithm $\mathcal{S}$ takes as input a message and the secret key, and outputs a signature σ. And the verification algorithm $\mathcal{V}$ takes as input a message, a signature and a public key, and returns 1 (if the signature is valid) or 0 (if not).

A proxy signature scheme $Pro\_Sig = (\mathcal{G}, \mathcal{K}, \mathcal{S}, \mathcal{V}, (\mathcal{D}, \mathcal{P}), \mathcal{PS}, \mathcal{PV}, \mathcal{ID})$ requires the presence of at least two users (user i delegates its signing capability to user j). The algorithms $\mathcal{G}$, $\mathcal{K}$, $\mathcal{S}$ and $\mathcal{V}$ are the same as explained above. The rest of protocols work as follows:

• $(\mathcal{D}, \mathcal{P})$ is a pair of (possibly interactive) algorithms, where user i delegates his signing capabilities to user j (proxy). The algorithm $\mathcal{D}$ takes as input the

public keys pk_i and pk_j and the secret key sk_i of the delegator. The algorithm $\mathcal{P}$ takes as input the public keys pk_i and pk_j and the secret key sk_j of the proxy signer. As a result of this interaction, the proxy signer (user j) obtains a proxy secret key skp_{ij} that he will use to sign messages on behalf of user i.

• The protocol $\mathcal{PS}$ is the proxy signing algorithm, which takes as input a proxy secret key skp and a message M, and outputs a proxy signature $p\sigma$. This proxy signature includes the public key of user j, the proxy signer.

• The protocol $\mathcal{PV}$ verifies the correctness of a proxy signature. It takes as input a message, a proxy signature and the public key of the original signer, and outputs 1 or 0.

• The proxy identification algorithm $\mathcal{ID}$ takes as input a valid proxy signature and outputs the identity of the proxy signer.

In order to analyze the security of such a proxy signature scheme, we must consider the most powerful attack against it; this adversary will try to forge a signature involving some honest user (say user 1). In order to do it, the adversary is allowed to corrupt all the users in a system except user 1; then, the adversary can request this user 1 to execute the different protocols of the scheme as many times as he wants, interacting with the corrupted users. Finally, the adversary tries to forge a new valid proxy signature computed by user 1 on behalf of a corrupted user, or by a corrupted user j on behalf of user 1 (provided user 1 has not been requested to delegate in user j, during the attack). A proxy signature scheme is secure if the probability of success of such an adversary is negligible.

This security model provided in [2] supports one level of delegation. A more complete formalization of the security of proxy signature schemes, given in [8], supports chains of more than one level of delegation. However, for simplicity, we will concentrate on the one-level model.

Triple Schnorr Proxy Signature Scheme. In [2], the authors explain some specific schemes which are proved to be secure according their security model. Now we explain one of these schemes, the triple Schnorr proxy signature scheme. We will refer to this scheme as $T\_Sch\_Pro\_Sig = (\mathcal{G}_{TS}, \mathcal{K}_{TS}, \mathcal{S}_{TS}, \mathcal{V}_{TS}, (\mathcal{D}_{TS}, \mathcal{P}_{TS}), \mathcal{PS}_{TS}, \mathcal{PV}_{TS}, \mathcal{ID}_{TS})$.

• The algorithms $(\mathcal{G}_{TS}, \mathcal{K}_{TS}, \mathcal{S}_{TS}, \mathcal{V}_{TS})$ are those of the standard Schnorr's signature scheme: $\mathcal{G}_{TS}$ generates the primes p and q, the element g and the hash functions H_1 and H_2. The algorithm $\mathcal{K}_{TS}$ generates secret key x and public key $y = g^x \bmod p$. The standard signing algorithm $\mathcal{S}_{TS}$ outputs a signature (r, s) on a message M. And $\mathcal{V}_{TS}$ verifies the correctness of the signatures. The main difference is that in order to sign a message M in a standard way, a user U_i must prepend a 1 to the message, and so apply $(r, s) = Sch\_Sig(1||M\ ,\ x_i\ ,\ H_1)$.

• The algorithms $(\mathcal{D}_{TS}, \mathcal{P}_{TS})$ are as follows. If a user U_i (with keys x_i and y_i) wants to delegate to a user U_j (with keys x_j and y_j), he creates a message ω which contains the information related to the delegation (identities of the original and proxy signers, dates of validity, which messages are allowed to be signed, etc.). Then user U_i computes the Schnorr signature $(r_i, s_i) = Sch\_Sig(0||y_i||y_j||\omega\ ,\ x_i\ ,\ H_1)$. User U_j verifies this signature and then com-

putes his proxy secret key as $skp_{ij} = (y_i||y_j||\omega\ ,\ r_i\ ,\ d_{ij})$, where $d_{ij} = s_i + x_j H_1(0||y_i||y_j||\omega\ ,\ r_i) \bmod q$. Note that the public key related to this secret key d_{ij} is $g^{d_{ij}} = r_i (y_i y_j)^{H_1(0||y_i||y_j||\omega\ ,\ r_i)} \bmod p$.

• To compute a proxy signature on a message M, on behalf of user U_i, user U_j employs his proxy secret key d_{ij} and hash function H_2 to compute the Schnorr signature $(r, s) = Sch\_Sig(0||M||y_i||y_j||\omega||r_i\ ,\ d_{ij}\ ,\ H_2)$. The final proxy signature is $p\sigma = (\omega, r_i, y_j, (r, s))$.

• To verify the correctness of a proxy signature $p\sigma = (\omega, r_i, y_j, (r, s))$ on a message M, where the original signer has public key y_i, the recipient must check the following equation (Schnorr verification with public key $g^{d_{ij}}$ and hash function H_2):

$$g^s = r \left[r_i (y_i y_j)^{H_1(0||y_i||y_j||\omega\ ,\ r_i)} \right]^{H_2(0||M||y_i||y_j||\omega||r_i\ ,\ r)} \bmod p\ .$$

• The proxy identification algorithm takes as input a proxy signature $p\sigma = (\omega, r_i, y_j, (r, s))$ and returns the identity which corresponds to the public key y_j.

Theorem 1. *If the discrete logarithm problem is hard, then the proxy signature scheme T_Sch_Pro_Sig is secure in the random oracle model.*

See [2] for the security model and the proof of this theorem.

2.2 Joint Generation of Discrete Logarithm Keys

In distributed public key cryptography, the secret tasks (decrypting or signing) are not performed by single users, but by entities formed by many users. Let $E = \{P^{(1)}, P^{(2)}, \ldots, P^{(n)}\}$ be a distributed entity formed by n participants. There is an *access structure* $\Gamma \subset 2^E$, which is formed by those subsets of participants which are authorized to perform the secret task. The access structure must be monotone increasing; that is, if $A_1 \in \Gamma$ is authorized, and $A_1 \subset A_2 \subset E$, then A_2 must be authorized, too.

The most usual strategy in distributed cryptography is to use *secret sharing schemes* (introduced in [1, 12]) to share the secret keys among the members of the entity. Some of these schemes do not need the presence of any trusted party (or dealer), and all the protocol can be performed by the members themselves. Linear secret sharing schemes, where the secret can be recovered as a linear combination of the shares from an authorized subset, are the most appropriate for being used as a component of distributed cryptographic protocols.

These distributed protocols must be secure in front of an attack of an adversary who corrupts a non-authorized subset of members of the entity. By corruption we mean that the adversary can see all the secret information of these users, and can control their behavior. The protocols are said to be *robust* if the dishonest members are always detected, and this fact does not avoid that the protocols finish in the correct way. In order to achieve robustness, *verifiable secret sharing schemes* are used.

A particular case of this kind of protocols is the joint generation of discrete logarithm keys. Each participant $P^{(\ell)} \in E$ obtains a secret value $x^{(\ell)} \in \mathbb{Z}_q$. These

values $\{x^{(\ell)}\}_{P^{(\ell)}\in E}$ form a sharing of the secret key $x \in \mathbb{Z}_q$, according to some linear secret sharing scheme realizing the access structure Γ. The corresponding public key $y = g^x \bmod p$ is made public, along with other values (commitments) which ensure the robustness of the protocol. We refer to an execution of this protocol as

$$(y, \{x^{(\ell)}\}_{P^{(\ell)}\in E}) = Jo\_DL\_KG(E, \Gamma) \ .$$

The details of this protocol can be found in [3] for the threshold case (that is, the access structure is $\Gamma = \{A \subset E \ : \ |A| \geq t\}$, for some threshold t) and in [4] for the case of general access structures.

Fact 1. The protocol Jo_DL_KG is simulatable.

This means that, given an adversary who corrupts a non-authorized subset B of members, there exists an algorithm SIM_1 which takes as input a public key $y \in \langle g \rangle$ and outputs values which are indistinguishable from those that the adversary would see in a real execution of the protocol Jo_DL_KG which would give y as the resulting public key. Mainly, the algorithm SIM_1 must simulate all the information which is made public in the protocol, and the secret information of the dishonest members in B.

2.3 Distributed Schnorr Signature Protocol

In a distributed signature scheme, a set E of users share the secret key of a standard signature scheme. If an authorized subset of members collaborate, they can produce a valid signature on a message. The recipient can verify the correctness of this signature, but cannot know if it has been generated in a standard or a distributed way.

These schemes are said to be *unforgeable* if an adversary who corrupts a non-authorized subset of members is not able to obtain a valid message-signature pair, even if the protocol is previously executed for other messages that the adversary adaptively chooses. The signing protocol is robust if the dishonest participants are detected and furthermore the output of the protocol is always a valid signature.

In the case of Schnorr's signature scheme, the threshold version was proposed in [13], and the version for general access structures was proposed in [4]. We consider the more general case with any access structure Γ. The scheme starts with the joint key generation, that is, an execution of $(y, \{x^{(\ell)}\}_{P^{(\ell)}\in E}) = Jo\_DL\_KG(E, \Gamma)$, and then a protocol to jointly sign a message. We refer to an execution of this last protocol as:

$$(r, s) = Dist\_Sch\_Sig(E, \Gamma, M, \{x^{(\ell)}\}_{P^{(\ell)}\in E}, H_1) \ ,$$

meaning that participants of entity E use their secret shares $\{x^{(\ell)}\}_{P^{(\ell)}\in E}$ of the secret key x (which have been distributed using a linear secret sharing scheme which realizes the access structure Γ), to jointly compute a standard Schnorr signature (r, s) of message M with hash function H_1. This implies that $g^s = ry^{H_1(M,r)} \bmod p$.

Fact 2. The protocol *Dist_Sch_Sig* is simulatable.

This fact means that, given an adversary who corrupts a non-authorized subset $B \notin \Gamma$ of participants, there is an algorithm SIM_2 which runs as follows: it takes as input (M, r, s), where (r, s) is a valid Schnorr signature for message M, along with all the information obtained by the adversary in the execution of the corresponding $(y, \{x^{(\ell)}\}_{P^{(\ell)} \in E}) = Jo\_DL\_KG(E, \Gamma)$. The output values are indistinguishable from those (public and secret information of the corrupted members) that the adversary would see in a real execution of the protocol $Dist\_Sch\_Sig(E, \Gamma, M, \{x^{(\ell)}\}_{P^{(\ell)} \in E}, H)$.

3 Fully Distributed Proxy Signature Schemes

In addition to individual proxy signature schemes, some distributed (usually threshold) proxy signature schemes have been proposed in the last years [15, 5]. In such schemes, a original signer delegates his capabilities to a proxy distributed entity. Members of an authorized subset of this entity can then jointly sign a message on behalf of the original signer. If the original signer is a distributed entity, too, then the proxy signature scheme is fully distributed [4].

As it has happened in the case of individual proxy signature schemes, no formal treatment of the security of distributed (and fully distributed) proxy signature schemes has been given until now. Some of the attacks which have been found against individual schemes are also applicable in the distributed versions of these schemes. For example, the attack explained in [6] against the individual proxy signature scheme in [7] can be also extended to an attack against the fully distributed proxy signature scheme in [4].

In this section, we formally define a fully distributed proxy signature scheme and the security requirements that such a scheme must satisfy. In some way, we extend the work done in [2] to the distributed scenario. Now there will be distributed entities $E_i = \{P_i^{(1)}, \ldots, P_i^{(n_i)}\}$ and $E_j = \{P_j^{(1)}, \ldots, P_j^{(n_j)}\}$ with their corresponding (monotone increasing) access structures $\Gamma_i \subset 2^{E_i}$ and $\Gamma_j \subset 2^{E_j}$. An authorized subset in Γ_i can delegate the signing capabilities of entity E_i to entity E_j. Then, an authorized subset in Γ_j can compute a proxy signature of entity E_j on behalf of entity E_i. Let us formalize the definition of all these protocols.

Let $Dist\_Sig = (\mathcal{G}, \mathcal{JKG}, \mathcal{DS}, \mathcal{V})$ be a distributed signature scheme. That is:

- The parameter-generator $\mathcal{G}$ takes as input a security parameter k and outputs some global (and public) parameters of the scheme (prime numbers, generators of the mathematical groups, etc.).
- The joint key generation protocol $\mathcal{JKG}$ is interactively performed by the members of each distributed entity E_i. It takes as input the global parameters and outputs a public key pk_i. Furthermore, each participant $P_i^{(\ell)} \in E_i$ obtains a secret share $sk_i^{(\ell)}$ of the secret key sk_i which matches with pk_i.

• The distributed signing algorithm $\mathcal{DS}$ takes as input a message and the secret shares of an authorized subset of members of the entity, and outputs a standard signature σ.

• The verification algorithm $\mathcal{V}$ takes as input a message, a signature and a public key, and returns 1 if the signature is valid, or 0 otherwise.

But a fully distributed proxy signature scheme $Dist\_Pro\_Sig = (\mathcal{G}, \mathcal{JKG}, \mathcal{DS}, \mathcal{V}, (\mathcal{DD}, \mathcal{DP}), \mathcal{DPS}, \mathcal{PV}, \mathcal{ID})$ requires also the following extra algorithms:

• $(\mathcal{DD}, \mathcal{DP})$ is a pair of (possibly interactive) algorithms. Entity E_i delegates its signing capabilities to entity E_j (proxy entity). The algorithm $\mathcal{DD}$ takes as input the public keys pk_i and pk_j and the shares of the secret key sk_i corresponding to some authorized subset of entity E_i. The algorithm $\mathcal{DP}$ takes as input the public keys pk_i and pk_j and the shares $\{sk_j^{(\ell)}\}_{P_j^{(\ell)} \in E_j}$ of the secret key of the proxy entity. As a result, each member $P_j^{(\ell)} \in E_j$ of the proxy entity obtains a share $skp_{ij}^{(\ell)}$ of the new proxy secret key skp_{ij}.

• The protocol $\mathcal{DPS}$ is the distributed proxy signing algorithm, which takes as input a message M and the shares of the proxy secret key skp_{ij} from some authorized subset of E_j, and outputs a proxy signature $p\sigma$. This proxy signature includes the public key pk_j of the proxy entity E_j.

• The protocol $\mathcal{PV}$ verifies the correctness of a proxy signature. It takes as input a message, a proxy signature and the public key of the delegator entity, and outputs 1 or 0.

• The proxy identification algorithm $\mathcal{ID}$ takes as input a valid proxy signature and outputs the identity of the proxy entity which has computed the signature.

3.1 Security Requirements

Intuitively, we want an adversary not to be able to forge a proxy or standard signature, even if he corrupts a non-authorized subset of each distributed entity which takes part in the system. In order to formally model this situation, we must consider a distributed entity E_1 and an adversary $\mathcal{DA}$ who corrupts a non-authorized subset $B_1 \subset E_1$, $B_1 \notin \Gamma_1$. The goal of the adversary is to forge a new proxy or standard signature realized by entity E_1 or on behalf of E_1.

Let $Dist\_Pro\_Sig = (\mathcal{G}, \mathcal{JKG}, \mathcal{DS}, \mathcal{V}, (\mathcal{DD}, \mathcal{DP}), \mathcal{DPS}, \mathcal{PV}, \mathcal{ID})$ be a fully distributed proxy signature scheme. We are going to consider an attack (or experiment) $\mathbf{D\_Exp}^{\mathcal{DA}}_{Dist\_Pro\_Sig}(k)$ performed by the adversary $\mathcal{DA}$ against the scheme $Dist\_Pro\_Sig$ under security parameter k.

The experiment starts with the generation of the global parameters. Adversary $\mathcal{DA}$ chooses the subset $B_1 \subset E_1$, such that $B_1 \notin \Gamma_1$, that he corrupts. Then the joint key generation protocol $\mathcal{JKG}$ is executed by the members of E_1 (here $\mathcal{DA}$ obtains the public key pk_1, all the information made public during the execution of the protocol, and the secret key shares $\{sk_1^{(b)}\}_{P_1^{(b)} \in B_1}$ of the corrupted participants).

The adversary initializes a counter $m = 1$, an empty set $Prox = \emptyset$ and an empty array $Array_{skp}^{(1)}$.

What can $\mathcal{DA}$ do? During the experiment, the adversary $\mathcal{DA}$ is allowed to execute

1. $\mathcal{DA}$ **registers** E_i**.** $\mathcal{DA}$ can create and register a new distributed entity E_i, for $i = m + 1$. The adversary controls the behavior of all the members of this entity. These members run the protocol $\mathcal{JKG}$ which produces a public key pk_i and shares of the corresponding secret key sk_i. A new empty array $Array_{skp}^{(i)}$ is created. The counter is incremented, $m := m + 1$.
2. E_1 **delegates in** E_i**.** $\mathcal{DA}$ can interact with the whole entity E_1 running the protocol $\mathcal{DD}(pk_1, pk_i, \{sk_1^{(\ell)}\}_{P_1^{(\ell)} \in E_1})$, and himself playing the role of entity E_i, for some $i \in \{2, 3, \ldots, m\}$, running the protocol $\mathcal{DP}(pk_1, pk_i, \{sk_i^{(\ell)}\}_{P_i^{(\ell)} \in E_i})$. The set $Prox$ increases to $Prox \cup \{pk_i\}$ (this set contains the public keys of the entities in which entity E_1 delegates during the experiment).
3. E_i **delegates in** E_1**.** $\mathcal{DA}$ can interact with entity E_1 running the protocol $\mathcal{DP}(pk_i, pk_1, \{sk_1^{(\ell)}\}_{P_1^{(\ell)} \in E_1})$, and himself playing the role of entity E_i, for some $i \in \{2, 3, \ldots, m\}$, running the protocol $\mathcal{DD}(pk_i, pk_1, \{sk_i^{(\ell)}\}_{P_i^{(\ell)} \in E_i})$. As a result, each participant $P_1^{(\ell)}$ of entity E_1 will obtain a share $skp_{i1}^{(\ell)}$ of the new proxy secret key. Note that the adversary knows the shares of the corrupted players in B_1. The whole set of shares $SKP_{i1} = \{skp_{i1}^{(\ell)}\}_{P_i^{(\ell)} \in E_i}$ is stored in the first available position of $Array_{skp}^{(i)}$. This array will therefore contain all the secret proxy keys corresponding to delegations of entity E_i into entity E_1. Obviously, the adversary has not full access to these arrays (he only knows the shares of the corrupted players in B_1).
4. E_1 **delegates in** E_1**.** $\mathcal{DA}$ can request that entity E_1 run the delegation protocol with itself. The adversary will see all the public information and the private information held by the corrupted players. As in Action 3, the shares of the resulting secret proxy key, SKP_{11}, are stored in the first available position of $Array_{skp}^{(1)}$.
5. **Standard distributed signature by** E_1**.** $\mathcal{DA}$ can ask the members of E_1 for executing the protocol $\mathcal{DS}$ for signing the message M that he chooses. He obtains all public information and private information of the dishonest players (in B_1).
6. **Distributed proxy signature by** E_1 **on behalf of** E_i**.** $\mathcal{DA}$ can request that members of E_1 use the shares of some of the proxy secret keys obtained from a delegation of entity E_i (Action 3), and which are stored in some position of $Array_{skp}^{(i)}$, to execute the protocol $\mathcal{DPS}$ with a message M that he chooses. Again, he obtains the signature, all the broadcast information and the private information of the corrupted players.

When is $\mathcal{DA}$ successful? Once the adversary has done these actions as many times as he wants, he eventually outputs a forgery of a standard signature (M, σ) or of a proxy signature $(M, p\sigma, pk)$.

- If (M, σ) satisfies $\mathcal{V}(M, \sigma, pk_1) = 1$, and M was not queried by $\mathcal{DA}$ to be signed as a standard distributed signature by entity E_1 (action 5), then the output of the experiment is 1 (successful forgery of a standard signature by entity E_1).
- If $(M, p\sigma, pk)$ satisfies $pk = pk_i$ for some $i \in \{1, 2, \ldots, m\}$, and $\mathcal{PV}(M, p\sigma, pk_i) = 1$, and $\mathcal{ID}(p\sigma) = pk_1$, and message M was not queried to be signed by E_1 on behalf of E_i (action 6), then the output of the experiment is 1 (successful forgery of a proxy signature by entity E_1 on behalf of some entity E_i).
- If $(M, p\sigma, pk)$ satisfies $pk = pk_1$, and $\mathcal{PV}(M, p\sigma, pk_1) = 1$, and $\mathcal{ID}(p\sigma) \notin Prox \cup \{pk_1\}$, then the output of the experiment is 1 (successful forgery of a proxy signature by some entity $E_i \neq E_1$, which was not designated by entity E_1 during the experiment, on behalf of entity E_1).

Otherwise, the output of the experiment $\mathbf{D\_Exp}^{\mathcal{DA}}_{Dist\_Pro\_Sig}(k)$ is 0. We define the probability of success of the adversary $\mathcal{DA}$ as the probability that the output of the experiment is 1. That is:

$$\mathbf{Succ}^{\mathcal{DA}}_{Dist\_Pro\_Sig}(k) = \Pr\left[\mathbf{D\_Exp}^{\mathcal{DA}}_{Dist\_Pro\_Sig}(k) \;=\; 1\right] .$$

Definition 1. *We say that a fully distributed proxy signature scheme Dist_Pro_Sig is secure if, for all polynomial time adversary $\mathcal{DA}$, we have that $\mathbf{Succ}^{\mathcal{DA}}_{Dist\_Pro\_Sig}(k)$ is negligible in the security parameter k.*

We recall that a function $f(k)$ is negligible in k if for all polynomial $p()$, there exists $k_p \in \mathbb{N}$ such that $f(k) \leq \frac{1}{p(k)}$, for all $k \geq k_p$.

This security model for fully distributed proxy signature schemes is the natural extension of the security model defined in [2] for individual proxy signature schemes (there, the adversary attacks a user; here, the adversary attacks an entity where he has corrupted some of its members).

4 A New Scheme

We now explain the natural way of fully distributing the triple Schnorr proxy signature scheme given in [2]. We follow the notation introduced in Section 3. We denote the fully distributed triple Schnorr proxy signature scheme by $T\_Sch\_Dist\_Pro\_Sig = (\mathcal{G}_{TS}, \mathcal{JKG}_{TS}, \mathcal{DS}_{TS}, \mathcal{V}_{TS}, (\mathcal{DD}_{TS}, \mathcal{DP}_{TS}), \mathcal{DPS}_{TS}, \mathcal{PV}_{TS}, \mathcal{ID}_{TS})$. The different protocols work as follows:

• The parameter generator $\mathcal{G}_{TS}$ takes as input a security parameter k and outputs the prime numbers p and q such that $q|p-1$, an element g with order q in $\mathbb{Z}_p^*$, and two hash functions $H_1,\ H_2 : \{0,1\}^* \to \mathbb{Z}_q$.

• The key generator $\mathcal{K}_{TS}$ for an entity E_i with access structure Γ_i consists of running the protocol $(y_i, \{x_i^{(\ell)}\}_{P_i^{(\ell)} \in E_i}) = Jo\_DL\_KG(E_i, \Gamma_i)$ for joint generating a public key and shares of the matching secret key (see Section 2.2).

• The distributed signature protocol $\mathcal{DS}_{TS}$ applied to a message M consists of prepending a 1 to the message and executing the protocol of joint computation

of a Schnorr signature $(r,s) = Dist\_Sch\_Sig(E_i, \Gamma_i,\ 1||M\ , \{x_i^{(\ell)}\}_{P_i^{(\ell)} \in E_i}, H_1)$ (see Section 2.3).

• The verification protocol $\mathcal{V}_{TS}$ verifies that (r,s) is a valid Schnorr signature for message $1||M$.

• The protocols $(\mathcal{DD}_{TS}, \mathcal{DP}_{TS})$ are as follows. If an entity E_i (with keys x_i and y_i, where x_i is shared) wants to delegate to an entity E_j (with keys x_j and y_j, where x_j is shared), members of entity E_i create a message ω which contains the information related to the delegation. Then members of E_i jointly compute the Schnorr signature

$$(r_i, s_i) = Dist\_Sch\_Sig(E_i, \Gamma_i,\ 0||y_i||y_j||\omega\ , \{x_i^{(\ell)}\}_{P_i^{(\ell)} \in E_i}, H_1)\ .$$

Each member $P_j^{(\ell)}$ of entity E_j verifies this signature and then computes his share of the proxy secret key as $skp_{ij}^{(\ell)} = (y_i||y_j||\omega\ ,\ r_i\ ,\ d_{ij}^{(\ell)})$, where

$$d_{ij}^{(\ell)} = s_i + x_j^{(\ell)} H_1(0||y_i||y_j||\omega\ ,\ r_i) \bmod q\ .$$

Note that the secret matching with the shares $\{d_{ij}^{(\ell)}\}_{P_j^{(\ell)} \in E_j}$ is the proxy secret key $d_{ij} = s_i + x_j H_1(0||y_i||y_j||\omega\ ,\ r_i) \bmod q$; and the public key related to this secret key d_{ij} is $g^{d_{ij}} = r_i (y_i y_j)^{H_1(0||y_i||y_j||\omega\ ,\ r_i)} \bmod p$.

• The protocol $\mathcal{DPS}_{TS}$ works as follows: to jointly compute a proxy signature on a message M, on behalf of entity E_i, members of the entity E_j employ their shares of the proxy secret key d_{ij} and the hash function H_2 to compute the Schnorr signature

$$(r,s) = Dist\_Sch\_Sig(E_j, \Gamma_j,\ 0||M||y_i||y_j||\omega||r_i\ , \{d_{ij}^{(\ell)}\}_{P_j^{(\ell)} \in E_j}, H_2)\ .$$

The final proxy signature is $p\sigma = (\omega, r_i, y_j, (r,s))$.

• To verify (protocol $\mathcal{PV}_{TS}$) the correctness of a proxy signature $p\sigma = (\omega, r_i, y_j, (r,s))$ on a message M, where the original signer entity has public key y_i, the recipient must check the following equation (Schnorr verification with public key $g^{d_{ij}}$ and hash function H_2):

$$g^s = r \left[r_i (y_i y_j)^{H_1(0||y_i||y_j||\omega\ ,\ r_i)} \right]^{H_2(0||M||y_i||y_j||\omega||r_i\ ,\ r)} \bmod p\ .$$

• The proxy identification algorithm $\mathcal{ID}_{TS}$ takes as input a proxy signature $p\sigma = (\omega, r_i, y_j, (r,s))$ and returns the entity whose public key is y_j.

4.1 Length of the Signatures

Let us consider the trivial solution to the proxy signature problem that we mentioned in the Introduction. A distributed proxy signature on a message M computed by an entity E_j on behalf of an entity E_i would consist in a tuple $\big(\omega, (r_i, s_i), M, (r,s)\big)$, where ω is the delegation message, (r_i, s_i) is the Schnorr's signature on ω computed by members of E_i in a distributed way, and (r,s) is the

Schnorr's signature on message M computed by members of E_j in a distributed way.

On the other hand, if we consider the proxy signatures which result from the scheme described in this section, they have the form $p\sigma = \big(\omega, r_i, M, (r, s)\big)$. Note that we have erased the term y_j corresponding to the public key of the proxy signer, because this information can be included in ω or in M. We can see that these signatures are shorter than in the trivial solution, because the term s_i from the Schnorr's signature on ω is not needed at all. Therefore, our solution is more efficient than the trivial one.

4.2 Security Analysis

The following theorem asserts that this fully distributed proxy signature scheme is secure in the model introduced in Section 3.1. We prove this fact by reduction to the security of the individual triple Schnorr scheme in the security model for individual proxy signatures (see Section 2.1).

Theorem 2. *If the discrete logarithm problem is hard, then the fully distributed proxy signature scheme T_Sch_Dist_Pro_Sig is secure in the random oracle model.*

Proof. Let us assume there exists an adversary $\mathcal{DA}$ against this fully distributed proxy signature scheme such that its success probability $\mathbf{Succ}^{\mathcal{DA}}_{T\_Sch\_Dist\_Pro\_Sig}(k)$ is non-negligible. We can then construct an adversary $\mathcal{A}$ and an experiment $\mathbf{Exp}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k)$ against the individual scheme $T\_Sch\_Pro\_Sig$, following the definition and notation of [2], as follows:

The public parameters (p, q, g, H_1, H_2) are generated, along with a public and secret key pair (x_1, y_1) for user U_1, where $y_1 = g^{x_1} \bmod p$. A counter m is initialized to 1, an empty set $Prox$ and an empty array $Array^{(1)}_{skp}$ are created. The value y_1 is given to the adversary $\mathcal{A}$.

Now $\mathcal{A}$ executes SIM_1 (see Fact 1) with input y_1 and the information related to the adversary $\mathcal{DA}$ (entity E_1, access structure Γ_1, set B_1 of corrupted players...). Therefore $\mathcal{A}$ obtains values which are indistinguishable from those that $\mathcal{DA}$ would have seen in a real execution of $K_{TS} = Jo\_DL\_KG$ which would have produced y_1 as the resulting public key.

Then, $\mathcal{A}$ requests $\mathcal{DA}$ to run the experiment $\mathbf{D\_Exp}^{\mathcal{DA}}_{T\_Sch\_Dist\_Pro\_Sig}(k)$. For that, $\mathcal{A}$ must provide $\mathcal{DA}$ with the information obtained from SIM_1 in the previous step, and also simulate the real environment of $\mathcal{DA}$ during the experiment $\mathbf{D\_Exp}^{\mathcal{DA}}_{T\_Sch\_Dist\_Pro\_Sig}(k)$, replying all its queries and actions:

1. If $\mathcal{DA}$ wants to register a new entity E_i, where $i = m+1$, then $\mathcal{A}$ registers a new user U_i (he is allowed to do so, see the security model in [2]), obtaining a pair (x_i, y_i). Then $\mathcal{A}$ executes SIM_1 with input y_i and gives the outputs to $\mathcal{DA}$. The counter is incremented, $m := m+1$, and an empty array $Array^{(i)}_{skp}$ is created.
2. When $\mathcal{DA}$ requires entity E_1 to delegate to entity E_i (with delegation message ω), then $\mathcal{A}$ requires user U_1 to delegate to user U_i. Therefore, $\mathcal{A}$ obtains

a valid Schnorr signature, under public key y_1 and hash function H_1, of the message $0||y_1||y_i||\omega$. Then $\mathcal{A}$ executes SIM_2 (see Fact 2) with input this pair message-signature and the information obtained in the first execution (with input y_1) of SIM_1. The output of SIM_2 perfectly simulates the view of $\mathcal{DA}$ during these queries. The set $Prox$ increases to $Prox \cup \{pk_i\}$.

3. When $\mathcal{DA}$ requires some entity E_i to delegate to entity E_1, then $\mathcal{A}$ requires user U_i to delegate to user U_1. If the delegation message is ω, then $\mathcal{A}$ obtains a valid Schnorr signature (r_i, s_i), under public key y_i and hash function H_1, of the message $0||y_i||y_1||\omega$. Now $\mathcal{A}$ executes SIM_2 for this pair message-signature. Furthermore, for all corrupted player $P_1^{(b)} \in B_1 \subset E_1$, $\mathcal{A}$ computes the corresponding share
$$d_{i1}^{(b)} = s_i + x_1^{(b)} H_1(0||y_i||y_1||\omega\ ,\ r_i) \bmod q$$
of the new proxy secret key, where $x_1^{(b)}$ are the shares of the secret key of entity E_1, obtained in the first execution of SIM_1. In this way, $\mathcal{A}$ simulates in a perfect way the view of $\mathcal{DA}$ for these queries. The first available position of $Array_{skp}^{(i)}$ is filled with these shares $d_{i1}^{(b)}$ and other random shares for the non-corrupted players (since $\mathcal{DA}$ has not full access to these arrays, it is not important what is put in the places corresponding to the non-corrupted players).
4. When $\mathcal{DA}$ requires entity E_1 to delegate to itself, $\mathcal{A}$ requires user U_1 to designate himself. If the delegation message is ω, then $\mathcal{A}$ obtains a valid Schnorr signature (r_1, s_1), under public key y_1 and hash function H_1, of message $0||y_1||y_1||\omega$. Then $\mathcal{A}$ executes SIM_2 for this pair message-signature. Again, for all corrupted players $P_1^{(b)} \in B_1 \subset E_1$, $\mathcal{A}$ computes the corresponding share
$$d_{11}^{(b)} = s_1 + x_1^{(b)} H_1(0||y_1||y_1||\omega\ ,\ r_1) \bmod q$$
of the new proxy secret key. These values and the output of SIM_2 perfectly simulate the view of $\mathcal{DA}$ during these queries. The next available position of $Array_{skp}^{(1)}$ is filled with the computed shares for the corrupted players in B_1 and with random numbers for the non-corrupted players.
5. When $\mathcal{DA}$ requires E_1 to compute a distributed Schnorr signature on a message M, $\mathcal{A}$ queries user U_1 to compute a Schnorr signature on message M (the same message and the same public key). The resulting signature and the message are given as inputs to SIM_2. The outputs simulate the view of $\mathcal{DA}$ during the execution of the distributed Schnorr signature protocol.
6. If $\mathcal{DA}$ requires entity E_1 to compute a proxy signature of message M on behalf of entity E_i (which has previously delegated to E_i by publishing a signature (r_i, s_i) on a delegation message ω), then $\mathcal{A}$ requires user U_1 to compute a proxy signature of message M on behalf of user U_i (who, of course, has previously delegated to U_1 by publishing exactly the signature (r_i, s_i) on the delegation message ω). The result is a valid Schnorr signature (r, s), of message $0||M||y_i||y_1||\omega||r_i$, under hash function H_2 and public key
$$r_i (y_i y_1)^{H_1(0||y_i||y_1||\omega\ ,\ r_i)}\ .$$

Then $\mathcal{A}$ can execute SIM_2 with input this message-signature pair, along with other information which $\mathcal{A}$ had obtained when E_i performed the considered delegation on E_1 (for example, the shares $d_{i1}^{(\ell)}$ of the corresponding secret proxy key). The output of SIM_2 simulates the view of $\mathcal{DA}$ in this phase of the experiment.

By assumption, and since $\mathcal{A}$ perfectly simulates the environment of $\mathcal{DA}$, one of the following facts happens with non-negligible probability:

• $\mathcal{DA}$ outputs $(M,(r,s))$ satisfying $\mathcal{V}_{TS}(M,(r,s),y_1)=1$, such that M was not queried by $\mathcal{DA}$ to be signed as a standard distributed signature by entity E_1 (action 5). Therefore, $\mathcal{A}$ did not query user U_1 to sign message M in the standard way, either, and so the output of the experiment $\mathbf{Exp}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k)$, performed by $\mathcal{A}$, would be 1 (successful forgery of a standard signature).

• $\mathcal{DA}$ outputs a forgery of a proxy signature by entity E_1, on behalf of entity E_i, of a message that was not queried by $\mathcal{DA}$ to be signed by E_1 on behalf of E_i during the experiment. Therefore, $\mathcal{A}$ obtains a forgery of a proxy signature by U_1, on behalf of U_i, of a message that $\mathcal{A}$ did not query user U_1 to sign on behalf of U_i. That is, the output of $\mathbf{Exp}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k)$ would be again 1.

• $\mathcal{DA}$ outputs a forgery of a proxy signature by some entity $E_i \neq E_1$ (which was not designated by entity E_1 at any time during the experiment) on behalf of entity E_1. Analogously, user U_i was never designated by user U_1 during the experiment performed by $\mathcal{A}$, but $\mathcal{A}$ obtains a valid proxy signature by user $U_i \neq U_1$ on behalf of user U_1. Thus, the output of $\mathbf{Exp}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k)$ would be 1.

Summing up, we have $\mathbf{Succ}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k) \geq \mathbf{Succ}^{\mathcal{DA}}_{T\_Sch\_Dist\_Pro\_Sig}(k)$. But we are assuming that $\mathbf{Succ}^{\mathcal{DA}}_{T\_Sch\_Dist\_Pro\_Sig}(k)$ is non-negligible. So we could conclude that $\mathbf{Succ}^{\mathcal{A}}_{T\_Sch\_Pro\_Sig}(k)$ is also non-negligible, which contradicts Theorem 1. Therefore, we prove that there can not exist an adversary $\mathcal{DA}$ with non-negligible probability of successfully attacking the scheme $T\_Sch\_Dist\_Pro\_Sig$, and so this scheme is provably secure (in the random oracle model, as it is the individual triple Schnorr scheme). This completes the proof. □

5 Conclusion

In this work we have taken one more step in the formalization of proxy signature schemes, by giving a security model for fully distributed proxy signature schemes. This new model is the natural extension of the security model introduced in [2] for individual proxy signature schemes with one level of delegation.

Furthermore, we present a fully distributed proxy signature scheme which is proved to be secure in the new model. The scheme is the distributed version of a individual scheme proposed in [2].

There are a lot of proxy signature schemes (individual or distributed) in the literature whose security has not been formally proved yet. We think that the two above-mentioned models, the one in [2] for individual proxy signature schemes and the one in this work for distributed schemes, along with the model in [8] for schemes with several levels of delegation, should be considered from now on, in order to prove the security of both existing and future schemes.

References

1. G.R. Blakley. Safeguarding cryptographic keys. *Proceedings of AFIPS'79*, pp. 313–317 (1979).
2. A. Boldyreva, A. Palacio and B. Warinschi. Secure proxy signature schemes for delegation of signing rights. Manuscript available at `http://eprint.iacr.org/2003/096/`
3. R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Proceedings of Eurocrypt'99*, LNCS **1592**, pp. 295–310 (1999).
4. J. Herranz and G. Sáez. Verifiable secret sharing for general access structures, with application to fully distributed proxy signatures. *Proceedings of Financial Cryptography Conference 2003*, LNCS **2742**, pp. 286–302 (2003).
5. S. Kim, S. Park and D. Won. Proxy signatures, revisited. *Proceedings of ICISC'97*, pp. 223–232 (1997).
6. J.Y. Lee, J.H. Cheon and S. Kim. An analysis of proxy signatures: is a secure channel necessary? *Proceedings of CT-RSA Conference 2003*, LNCS **2612**, pp. 68–79 (2003).
7. B. Lee, H. Kim and K. Kim. Strong proxy signature and its applications. *Proceedings of SCIS'01*, Vol. 2/2, pp. 603–608 (2001).
8. T. Malkin, S. Obana and M. Yung. The hierarchy of key evolving signatures and a characterization of proxy signatures. *Proceedings of Eurocrypt'04*, LNCS **3027**, Springer-Verlag, pp. 306-322 (2004).
9. M. Mambo, K. Usuda and E. Okamoto. Proxy signatures: delegation of the power to sign messages. *IEICE Transactions Fundamentals*, Vol. E79-A, No. **9**, pp. 1338–1353 (1996).
10. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, Vol. **13**, Num. **3**, pp. 361–396 (2000).
11. C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, Vol. **4**, pp. 161–174 (1991).
12. A. Shamir. How to share a secret. *Communications of the ACM*, No. **22**, pp. 612–613 (1979).
13. D.R. Stinson and R. Strobl. Provably secure distributed Schnorr signatures and a (t, n) threshold scheme for implicit certificates. *Proceedings of ACISP'01*, LNCS **2119**, Springer-Verlag, pp. 417–434, (2001).
14. H.M. Sun and B.T. Hsieh. On the security of some proxy signature schemes. Manuscript available at `http://eprint.iacr.org/2003/068/` (2003).
15. K. Zhang. Threshold proxy signature scheme. *Proceedings of the 1997 Information Security Workshop, Japan*, pp. 191–197 (1997).

New ID-Based Threshold Signature Scheme from Bilinear Pairings*

Xiaofeng Chen[1], Fangguo Zhang[1], Divyan M. Konidala[2], and Kwangjo Kim[2]

[1] School of Information Science and Technology,
Sun Yat-sen University, Guangzhou 510275, P.R. China
{isschxf, isdzhfg}@zsu.edu.cn
[2] International Research center for Information Security (IRIS),
Information and Communications University(ICU),
103-6 Munji-dong, Yusong-ku, Taejon, 305-714 Korea
{divyan, kkj}@icu.ac.kr

Abstract. ID-based public key systems allow the user to use his/her identity as the public key, which can simplify key management procedure compared with CA-based public key systems. However, there is an inherent disadvantage in such systems: the problem of private key escrow, *i.e.*, the "trusted" Private Key Generator (PKG) can easily impersonate any user at any time without being detected. Although the problem of escrowing the private key may be reduced by distributing the trust onto multiple centers, it will decrease the efficiency of the systems. Chen *et al.* first proposed a novel ID-based signature scheme without trusted PKG from bilinear pairings [10], *i.e.*, there is only one PKG who is not assumed to be honest in their scheme. However, the signature scheme cannot be extended to a threshold one. In this paper we propose another ID-based signature scheme without trusted PKG from bilinear pairings. Moreover, we propose an ID-based threshold signature scheme without trusted PKG, which simultaneously overcomes the problem of key escrow and adopts the approach that the private key associated with an identity rather than the master key of PKG is shared.

Keywords: ID-based threshold signature, Bilinear pairings, Key escrow.

1 Introduction

The idea of threshold cryptography is to distribute the secret information (*i.e.*, a secret key) and computation (*i.e.*, decryption or signature generation) among multi parties in order to prevent a single point of failure or abuse. For example, let Alice be the president of a committee, she shared her power of signing (or decrypting) among a number of servers in such a way that only more than a certain

* This work was supported by a grant No.R12-2003-004-01004-0 from the Ministry of Science and Technology, Korea and the National Natural Science Foundation of China (No. 60403007).

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 371–383, 2004.

number of secret shares can be used to sign a message or decrypt a ciphertext on behalf of her. There are plenty of research on threshold cryptographic schemes under CA-based public key setting [6, 13, 21, 24].

In 1984, Shamir [22] introduced the concept of ID-based systems, which simplifies key management procedure of CA-based PKI. The idea of ID-based systems is that the identity information of the user $\mathcal{I}$ acts as his/her public key $\mathcal{P}$, and a trusted third party, called Private Key Generator (PKG), calculates the private key for the user. ID-based systems can be a good alternative for CA-based systems from the viewpoint of efficiency and convenience.

The bilinear pairings, namely the Weil pairing and the Tate pairing of algebraic curves, are important tools for research on algebraic geometry. The use of them in cryptography goes back to the results of Menezes-Okamoto-Vanstone [19] and Frey-Rück [11]. However, their works were to attack elliptic curve or hyperelliptic curve cryptosystems (*i.e.*, using pairings to transform the ECDLP or HCDLP into a discrete logarithm problem in the multiplicative group of a finite field). During the last couple of years, the bilinear pairings have initiated some completely new fields in cryptography, making it possible to realize cryptographic primitives that were previously unknown or impractical [4, 5]. More precisely, they are important tools for construction of ID-based cryptographic schemes [3, 4, 9, 17, 20, 23, 25].

However, there are some drawbacks in ID-based systems [9, 14, 17]. The most criticism against ID-based systems is that PKG knows the private key of all users, so he is able to impersonate any user to sign a document or decrypt an encrypted message. It implies that the PKG must be trusted unconditionally otherwise the systems will soon be collapsed. However, it would be difficult to assume the existence of a trusted party in an *ad hoc* network, where the communication parties are changing frequently.

Boneh and Franklin [4] proposed that the threat from escrowing the private key could be reduced by using "distributed PKGs". On the other hand, they briefly mentioned that each PKG of the "distributed PKGs" can act as a decryption (similarly, a signature generation) server. However, it is a disadvantage in Boneh and Franklin's scheme for the PKG to be involved in the particular applications, which is opposed to the Shamir's original proposal that the service of the PKG is limited to issue private keys. The original purpose of "distributed PKGs" is to prevent a single dishonest PKG possessing the users' private key, rather than to distribute a user's private key. Libert and Quisquater [18] proposed a somewhat different method where one PKG plays a role as a dealer. However, the PKGs in such schemes are still involved in particular applications.

Until very recently, Baek and Zheng [1] suggested a new approach for ID-based threshold decryption in which the private key associated with an identity rather than the master key of PKG is shared. Moreover, they [2] first proposed an ID-based threshold signature scheme without distributed PKGs. However, it still suffers the problem of private key escrow as the traditional ID-based systems. Though the scheme [2] can incorporate the distributed PKGs techniques to solve the key escrow problem, we argue that using distributed PKGs will

increase the communication and computation cost of the systems. To the best of our knowledge, there seems no ID-based threshold signature scheme without distributed PKGs which simultaneously overcomes the problem of key escrow and adopts the approach that the private key associated with an identity rather than the master key of PKG is shared.

Recently, a novel ID-based signature without the trusted PKG from bilinear pairings [10] is proposed. There is only one PKG who is not assumed to be trusted in the systems, which combines the advantages of both CA-based systems (no key escrow) and ID-based systems (no certificate) while removing their disadvantages. However, it seems difficult to extend the signature scheme to a threshold one. In this paper we propose another ID-based signature scheme without the trusted PKG from bilinear pairings. Moreover, we extend it to an ID-based threshold signature scheme without distributed PKGs which overcomes the problem of key escrow. Meanwhile, we adopt the approach that the private key associated with an identity rather than the master key of PKG is shared in the proposed scheme.

The rest of the paper is organized as follows: Some preliminaries are given in Section 2. Our new ID-based signature scheme from bilinear pairings is given in Section 3. The proposed ID-based threshold signature scheme is given in Section 4 and the analysis of our scheme is given in Section 5. Finally, concluding remarks will be made in Section 6.

2 Preliminaries

In this Section, we will briefly describe the basic definition and properties of bilinear pairings and gap Diffie-Hellman group. We also introduce ID-based public key setting and a knowledge proof for the equality of two discrete logarithm from bilinear pairings.

2.1 Bilinear Pairings

Let G_1 be a cyclic additive group generated by P, whose order is a prime q, and G_2 be a cyclic multiplicative group of the same order q. Let a and b be elements of Z_q^*. We assume that the discrete logarithm problem (DLP) in both G_1 and G_2 is hard. A bilinear pairing is a map $e : G_1 \times G_1 \rightarrow G_2$ with the following properties:

1. Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$;
2. Non-degenerate: There exists $P, Q \in G_1$ such that $e(P, Q) \neq 1$;
3. Computable: For all $P, Q \in G_1$, there is an efficient algorithm to compute $e(P, Q)$.

2.2 Gap Diffie-Hellman Group

Let G_1 be a cyclic additive group generated by P with the prime order q. Assume that the inversion and multiplication in G_1 can be computed efficiently. We introduce the following problems in G_1.

1. Discrete Logarithm Problem (DLP): Given two elements P and Q, to find an integer $n \in Z_q^*$, such that $Q = nP$ whenever such an integer exists.
2. Computation Diffie-Hellman Problem (CDHP): Given P, aP, bP for $a, b \in Z_q^*$, to compute abP.
3. Decision Diffie-Hellman Problem (DDHP): Given P, aP, bP, cP for $a, b, c \in Z_q^*$, to decide whether $c \equiv ab \bmod q$.

We call G_1 a gap Diffie-Hellman group if DDHP can be solved in polynomial time but there is no polynomial time algorithm to solve CDHP with non-negligible probability. Such group can be found in supersingular elliptic curve or hyperelliptic curve over finite field, and the bilinear pairings can be derived from the Weil or Tate pairings. For more details, see [4, 7, 12, 17].

In the following we always define G_1 be a gap Diffie-Hellman group of prime order q, G_2 be a cyclic multiplicative group of the same order q and a bilinear pairing $e : G_1 \times G_1 \to G_2$.

2.3 ID-Based Setting from Bilinear Pairings

The ID-based public key systems allow some public information of the user such as name, address and email *etc.*, rather than an arbitrary string to be used his public key. The private key of the user is calculated by PKG and sent to the user via a secure channel.

ID-based public key setting from bilinear pairings can be implemented as follows:

- **Setup:** PKG chooses a random number $s \in Z_q^*$ and set $P_{pub} = sP$. Define a cryptographic hash function $H_2 : \{0,1\}^* \to G_1$. The center publishes system parameters $params = \{G_1, G_2, e, q, P, P_{pub}, H_2\}$, and keep s as the *master-key*, which is known only himself.
- **Extract:** A user submits his/her identity information ID to PKG. PKG computes the user's public key as $Q_{ID} = H_2(ID)$, and returns $S_{ID} = sQ_{ID}$ to the user as his/her private key.

2.4 ID-Based Knowledge Proof for the Equality of Two Discrete Logarithm from Bilinear Pairings

A prover with possession a secret number $\beta \in Z_q$ wants to show that $\log_g u = \log_h v$ while without exposing β, where $u = g^\beta$, $v = h^\beta$. Chaum and Pedersen [8] first proposed an interactive protocol to solve this problem. Motivated by this idea, Baek and Zheng [1, 2] construct a new ID-based knowledge proof for the equality of two discrete logarithm from bilinear pairings.

Define $g = e(P, Q_{ID})$, $u = e(P_{pub}, Q_{ID})$, $h = e(L, Q_{ID})$ and $v = e(L, S_{ID})$, where P and L are independent points of G_1. The following protocol presents a knowledge proof of that $\log_g u = \log_h v$. An interesting property of this proof is that even the prover does not know the discrete logarithm $\log_g u = \log_h v$ (just be convinced that it equals to the master-key s of the PKG), which is different from the previous protocols. With the notation of [5], $< g, h, u, v >$ is called a Diffie-Hellman tuple.

- The prover randomly chooses an element Q in G_1 and computes $a = e(P, Q)$, $b = e(L, Q)$. The prover sends (a, b) to the verifier.
- The verifier randomly chooses an integer $c \in Z_q$ and sends c to the prover.
- The prover computes $S = Q + cS_{ID}$ and sends S to the verifier.
- The verifier checks whether $e(P, S) = au^c$ and $e(L, S) = bv^c$. If both the equations hold, returns "*accept*"; else, returns "*reject*".

As claimed in [1, 2], the above protocol can be easily converted a non-interactive knowledge proof:

- The prover randomly chooses an element Q in G_1 and computes $a = e(P, Q)$, $b = e(L, Q)$.
- Let $c = H(a, b, h, v)$, the prover computes $S = Q + cS_{ID}$ and sends (a, b, S) to the verifier.
- The verifier computes $c = H(a, b, g, h)$ and checks whether $e(P, S) = au^c$ and $e(L, S) = bv^c$. If both the equations hold, returns "*accept*"; else, returns "*reject*".[1]

3 New ID-Based Signature Scheme Without Trusted PKG

In this section, we first present our new ID-based key setting from bilinear pairings, and then propose a concrete signature scheme without the trusted PKG to solve the problem of key escrow, *i.e.*, we do not use the distributed PKGs in our system and the single PKG is assumed no longer to be a trusted party.

Define three cryptographic hash functions $H_1 : \{0,1\}^* \times G_1 \rightarrow G_1$, $H_2 : \{0,1\}^* \rightarrow G_1$ and $H_3 : {G_2}^4 \rightarrow Z_q$.

3.1 New ID-Based Public Key Setting from Bilinear Pairings

[**Setup**]

PKG chooses a random $s \in Z_q^*$ and sets $P_{pub} = sP$. The public parameters of the system are $params = \{G_1, G_2, e, q, P, P_{pub}, H_1, H_2, H_3\}$. PKG keeps s secretly as the *master-key*.

[**Extract**]

A user submits his (or her) identity information ID and authenticates himself (or herself) to PKG. The user then randomly chooses an integer $r \in Z_q^*$ as his long-term private key and sends rP to PKG. PKG computes $S_{ID} = sQ_{ID} = sH_1(ID||t, rP)$ and sends it to the user via a secure channel, where t is the life span of r. The user's private key pair are S_{ID} and r and the public key is ID.

The user should update his key pair after period of t. For the sake of simplicity, we do not discuss this problem here.

[1] The prover also can send (c, S) to the verifier. The verifier computes $a' = e(P, S)/u^c$, $b' = e(L, S)/v^c$ and $c' = H(a', b', h, v)$. If $c = c'$, the verifier accepts the proof; else reject the proof. Therefore, the length of proof is decreased.

3.2 New ID-Based Signature Scheme from Bilinear Pairings

Chen *et al.* [10] have proposed an ID-based signature scheme without the trusted PKG based on Cha and Cheon's signature scheme [7]. But it is unsuitable for designing threshold signature scheme. Here we propose a new ID-based signature scheme without the trusted PKG and then extend it to a threshold scheme.

[**Signing**]

Suppose that the message to be signed is m and the signer's identity is ID.

- The signer computes $T = rH_2(m)$.
- The signer computes $v = e(H_2(m), S_{ID})$.
- Let $g = e(P, Q_{ID})$, $u = e(P_{pub}, Q_{ID})$ and $h = e(H_2(m), Q_{ID})$, the signer proves that (g, h, u, v) is a Diffie-Hellman tuple by using a non-interactive knowledge proof for the equality of two discrete logarithm. Let the proof be $(a = e(P, Q), b = e(H_2(m), Q), S = Q + cS_{ID})$, where Q is a randomly chosen element in G_1 and $c = H_3(a, b, h, v)$.

Then (T, v, rP) and the corresponding proof (a, b, S) is the signature of the message of m.

[**Verification**]

The verifier computes $Q_{ID} = H_1(ID||t, rP)$, $h = e(H_2(m), Q_{ID})$, $u = e(P_{pub}, Q_{ID})$, $c = H_1(a, b, h, v)$. He accepts the signature if the following equations hold:

$$e(T, P) = e(H_2(m), rP)$$
$$e(P, S) = au^c,\ e(H_2(m), S) = bv^c$$

3.3 Security Analysis of Our Scheme

Theorem 1. *The proposed ID-based signature scheme reaches Girault's trusted level 3.*

Proof. Suppose PKG wants to impersonate an honest user whose identity information is ID. He can do as follows:

- PKG randomly chooses $r' \in Z_q^*$ and computes $S_{ID'} = sH_1(ID||t, r'P)$.
- He then performs the above signing protocol for the message m.
- Output $(T', v', r'P, a', b', S')$.

Because $e(T', P) = e(H_2(m), r'P)$, $e(P, S') = a'u'^c$, and $e(H_2(m), S') = b'v'^c$, where $u' = e(P_{pub}, Q'_{ID})$, $c = H_3(a', b', e(H_2(m), Q'_{ID}), v')$, and $Q'_{ID} = H_1(ID||t, r'P)$, PKG successfully forged a "valid" signature of the target user for the message m.

However, the user can provide a proof to convince that the signature is forged by PKG, which is similar to CA-based systems.[2] He first sends rP to

[2] In the CA-based systems, CA also can forge a user's certificate and impersonate the user to communicate with others. However, the user can accuse the dishonest CA because there exist his two different "valid" certificates issued by the same CA.

the arbiter, and then provides a "knowledge proof" that he knows $S_{ID} = sH_1(ID||t, rP)$: the arbiter randomly chooses a secret integer $a \in Z_q$ and sends aP to the user; the user then computes $e(S_{ID}, aP)$. If the equation $e(S_{ID}, aP) = e(H_1(ID||t, rP), P_{pub})^a$ holds, *i.e.*, identity ID corresponds to rP and $r'P$ for a same period t, the arbiter deduces that PKG is dishonest because the *master-key* s is only known to him.

Therefore, our scheme reaches Girault's trusted lever 3 [16], *i.e.*, the authority does not know the private key of the users, and it can be proven that the authority generates false witness if he does so. □

Theorem 2. *In the random oracle, our signature scheme is existentially unforgeable against adaptively chosen message and ID attacks under the assumption of CDHP in G_1 is intractable.*

Proof. In our scheme, the partial signature T is the "real" signature of the user for the message. The knowledge proof (a, b, S) and v can be used to convince the verifier that rP correspondences to ID for the period t. We consider the following two kinds of adversaries:

Case 1: Active Adversary

Since PKG is not a trusted party, we consider that an active adversary can collude with PKG. For a randomly chosen target user whose identity is ID. The adversary can know the target user's long-term public key rP and partial private key S_{ID} from PKG. So, it is trivial for the adversary to generate v and the proof (a, b, S) for any message. If he can compute the corresponding V for a message m, he can successfully forge a signature of the user for the message m. We consider the following game:

Suppose the adversary can query to H_2 adaptively at most k times. Suppose the i-th input of query is m_i and he gets the corresponding signature T_i, here $1 \leq i \leq k$. Finally, he outputs a new pair (m, T). We say that the adversary wins the game if m is not queried and $e(T, P) = e(H_2(m), rP)$.

If there exists an algorithm $\mathcal{A}_0$ for an adaptively chosen message attack to our scheme with a non-negligible probability, we can construct an algorithm $\mathcal{A}_1$ as follows:

- choose an integer $u \in \{1, 2, \cdots, k\}$. Define $\mathbf{Sign}(H_2(m_i)) = T_i$.
- For $i = 1, 2, \cdots, k$, $\mathcal{A}_1$ responds to $\mathcal{A}_0$'s queries to H_2 and **Sign**, while for $i = u$, $\mathcal{A}_1$ replaces m_u with m.
- $\mathcal{A}_0$ outputs (m_{out}, V_{out}).
- If $m_{out} = m$ and the signature T is valid, $\mathcal{A}_1$ outputs (m, T). Otherwise, outputs *Fail*.

Note that u is randomly chosen, $\mathcal{A}_0$ knows nothing from the queries result. Also, since H_2 is a random oracle, the probability that the output of $\mathcal{A}_0$ is valid without query of $H_2(m)$ is negligible. Let $H_2(m) = eP$, we obtain $T = reP$ from P, rP and eP, *i.e.*, we solved CDHP in G_1.

Actually, V can be regarded as the short signature of the message m and $(P, rP, H_2(m), T)$ is a valid Diffie-Hellman tuple. We know that the probability

of the adversary can successfully forge a valid signature is negligible. For more details, see reference [5].

Case 2: Passive Adversary

A passive adversary cannot collude with the PKG. In this case, for a target user whose identity is ID, the adversary cannot know the information of $S_{ID} = sH_1(ID||t, rP)$(*i.e.*, a "certificate" in CBE scheme [15]) from PKG. In the following we will prove that his success probability of forgery of a valid signature is negligible, which is similar to Cha-Cheon's proof [7].

As we mentioned above, an identity ID only corresponds to one unique rP for a period of time t, so (ID, rP) can be extracted at most once. Define q_{H_1} is the maximum number of queries to H_1. If there exists an algorithm $\mathcal{A}_0$ for an adaptively chosen message and ID attack to our scheme with a non-negligible probability, we can construct an algorithm $\mathcal{A}_1$ as follows:

- choose an integer $u \in \{1, 2, \cdots, q_{H_1}\}$. Define (ID_i, r_iP) the i-th input of query H_2.
- $\mathcal{A}_1$ responds to $\mathcal{A}_0$'s queries to H_1, H_2, H_3, **Extract**, and **Signing**, while for $i = u$, $\mathcal{A}_1$ replaces ID_u, r_uP with ID, rP.
- $\mathcal{A}_0$ outputs $(ID_{out}, r_{out}P, m, T, v, a, b, S)$.
- If $ID_{out} = ID$ and the signature is valid, $\mathcal{A}_1$ outputs $(ID, rP, m, T, v, a, b, S)$. Otherwise, outputs *Fail*.

Note that u is randomly chosen, $\mathcal{A}_0$ knows nothing from the queries result. Also, since H_1, H_2 and H_3 are random oracles, the probability that the output of $\mathcal{A}_0$ is valid without query of $H_1(ID||t, rP)$ is negligible. So, $\mathcal{A}_1$ can be used for an adaptively chosen message and given ID attack to our scheme with a non-negligible probability. We then use $\mathcal{A}_1$ to construct an algorithm $\mathcal{A}_2$ to solve CDHP in G_1:

- Given P, sP, lP and let $P_{pub} = sP$. Choose integers $x_i \in Z_q$ and let (ID_i, r_iP) the i-th input of query H. Define

$$H(ID_i||t, r_iP) = \begin{cases} lP, & \text{if } \ ID_i = ID \\ x_iP, & \text{otherwise} \end{cases}$$

- $\mathcal{A}_2$ responds to $\mathcal{A}_1$'s queries to H_1, H_2, H_3, **Extract**, and **Signing**.
- If $\mathcal{A}_1$ outputs a valid message-signature pair $(ID, rP, m, T, v, a, b, S)$, $\mathcal{A}_2$ then replays with the same random tape but a different choice of H_3, for example $H_3^{'}$. $\mathcal{A}_2$ outputs two valid message-signature pairs $(ID, rP, m, T, v, a, b, S)$ and $(ID, rP, m, T, v, a, b, S')$.

Note that $S = Q + cS_{ID}$ and $S' = Q + c'S_{ID}$, we have $S_{ID} = (c - c')^{-1}(S - S')$. Therefore, we can obtain $S_{ID} = slP$ from P, sP and lP, *i.e.*, we solved CDHP in G_1. □

4 ID-Based Threshold Signature Scheme Without Trusted PKG from Bilinear Pairings

Although the scheme [2] can incorporate the distributed PKGs, we argue that it will decrease the efficiency of the scheme to solve the key escrow problem by using distributed PKGs. In the following, based on the approach that the private key associated with an identity rather than the master key of PKG is shared, we propose an ID-based threshold signature scheme without distributed PKGs which overcomes the key escrow problem.

Private Key Distribution: *The public key setting is the same as above. Suppose the private key of the user with identity ID is r and S_{ID}. He distributes his private key to n servers as follows:*

- Chooses $a_i \in_R Z_q$ and $R_i \in_R G_1$ for $1 \leq i \leq t-1$.
- Let
$$h(x) = r + a_1 x + a_2 x^2 + \cdots + a_{t-1} x^{t-1}$$
$$H(x) = S_{ID} + xR_1 + x^2 R_2 + \cdots + x^{t-1} R_{t-1}$$
Computes the distributed private key $h(i) = r_i$, $H(i) = S_i$ and the corresponding verification key $l_i = r_i P$, $u_i = e(P, S_i)$ and then sends them to server Γ_i for $1 \leq i \leq n$. Note that $h(x) = \sum_{j \in \Phi} c_{xj}^{\Phi} r_j$ and $H(x) = \sum_{j \in \Phi} c_{xj}^{\Phi} S_j$, where $c_{xj}^{\Phi} = \prod_{l \in \Phi, l \neq j} \frac{x-l}{j-l}$, $\Phi \subset \{1, 2, \cdots, n\}$ be a set and $|\Phi| \geq t$.
- The server Γ_i verifies the validity of l_i, u_i and publishes them while keeps r_i, S_i secret.

Signing: *Each of $\{\Gamma_j\}_{j \in \Phi}$ performs the following to jointly create a signature for a message m.*

- Computes and broadcasts $T_j = r_j H_2(m)$.
- Computes and broadcasts $v_j = e(H_2(m), S_j)$.
- Computes and broadcasts $a_j = e(P, Q_j), b_j = e(H_2(m), Q_j)$, where Q_j is a randomly chosen element in G_1.
- Computes $a = \prod_{j \in \Phi} a_j^{c_{0j}^{\Phi}}$, $b = \prod_{j \in \Phi} b_j^{c_{0j}^{\Phi}}$, $v = \prod_{j \in \Phi} v_j^{c_{0j}^{\Phi}}$.
- Broadcasts $W_j = Q_j + cS_j$, where $c = H(a, b, h, v)$ and $h = e(H_2(m), Q_{ID})$.
- Each server $i \in \Phi$ checks whether $e(T_j, P) = e(H_2(m), l_j)$, $e(P, W_j) = a_j u_j^c$ and $e(H_2(m), W_j) = b_j v_j^c$ for $j \in \Phi$ and $j \neq i$. If the equations fails for some j, then broadcasts **Complaint** against server j.
- If all the servers are honest, computes $T = \sum_{j \in \Phi} c_{0j}^{\Phi} T_j$, $S = \sum_{j \in \Phi} c_{0j}^{\Phi} W_j$.

Then (T, v, rP) and the corresponding proof (a, b, S) is the signature of the message of m.

Verification: *The verifier first computes $Q = H_2(ID, rP)$, $h = e(H_2(m), Q_{ID})$, $u = e(P_{pub}, Q_{ID})$, $c = H_1(a, b, h, v)$. He accepts the signature if the following equations hold:*

$$e(T, P) = e(H_2(m), rP)$$
$$e(P, S) = au^c, \ e(H_2(m), S) = bv^c$$

5 Analysis of Our Threshold Signature Scheme

5.1 Correctness

Note that

$$T = \sum_{j \in \Phi} c_{0j}^{\Phi} T_j = \sum_{j \in \Phi} c_{0j}^{\Phi} r_j H_2(m) = r H_2(m)$$

$$S = \sum_{j \in \Phi} c_{0j}^{\Phi} W_j = \sum_{j \in \Phi} c_{0j}^{\Phi} (Q_j + cS_j)$$

Therefore, we have

$$e(T, P) = e(H_2(m), rP)$$

$$\begin{aligned} e(P, S) = &\ e(P, \sum_{j \in \Phi} c_{0j}^{\Phi} (Q_j + cS_j)) = e(P, \sum_{j \in \Phi} c_{0j}^{\Phi} Q_j + c \sum_{j \in \Phi} c_{0j}^{\Phi} S_j) \\ = &\ e(P, \sum_{j \in \Phi} c_{0j}^{\Phi} Q_j) e(P, S_{ID})^c = a u^c \end{aligned}$$

and

$$\begin{aligned} e(H_2(m), S) = &\ e(H_2(m), \sum_{j \in \Phi} c_{0j}^{\Phi} (Q_j + cS_j)) = e(H_2(m), \sum_{j \in \Phi} c_{0j}^{\Phi} Q_j + c \sum_{j \in \Phi} c_{0j}^{\Phi} S_j) \\ = &\ e(H_2(m), \sum_{j \in \Phi} c_{0j}^{\Phi} Q_j) e(H_2(m), S_{ID})^c = b v^c \end{aligned}$$

5.2 Robustness

Theorem 3. *The proposed ID-based threshold signature scheme is robust, i.e., the scheme outputs correctly even in the presence of a malicious adversary that makes the corrupted servers deviate from the normal execution.*

Proof. The robustness of "Private Key Distribution" is trivial for each servers can validate his private key share using the published verification key share.

In the "Signing" protocol, if all the following equations hold, the server Γ_j is sure not to be corrupted by a malicious adversary: $e(T_j, P) = e(H_2(m), l_j)$, $e(P, W_j) = a_j u_j^c$ and $e(H(m), W_j) = b_j v_j^c$. □

5.3 Security

Motivated by Gennaro *et al*'s idea for proving the security of the threshold DSS signature scheme, Baek and Zheng [2] defined "**Simulatability**" of the ID-based threshold signature and proved the relationship between the security of ID-based threshold signature and that of ID-based signature.

Definition 1. *An ID-based threshold signature scheme is said to be* simulatable *if the following conditions hold.*

1. "Private Key Distribution" is simulatable: Given the system parameters params and the identity ID, there exists a simulator which can simulate the view of the adversary on an execution of "Private Key Distribution".
2. "Signing" is simulatable: Given the system parameters params the identity ID, the message m, the corresponding signature σ, $t-1$ private key shares and the corresponding verification key shares, there is a simulator which can simulate the view of the adversary on an execution of " Signing".

Theorem 4. *If an ID-based threshold signature scheme is simulatable and the ID-based signature is secure in the sense of unforgeability, then the ID-based threshold signature scheme is also secure in the sense of unforgeability.*

Therefore, we only need to prove our ID-based threshold signature scheme is simulatable.

Lemma 1. *The proposed ID-based threshold signature scheme is simulatable.*

Proof. (sketch) Without loss of generality, we assume that the servers corrupted by the adversary are Γ_i, where $1 \leq i \leq t-1$. Firstly we prove "Private Key Distribution" is simulatable. Given the system parameters *params* and the identity ID, the adversary computes $u = e(P_{pub}, Q_{ID})$. Note that $u = \prod_{j=1}^{t} u_i^{c_{0j}^{\Phi}}$, so the adversary can compute $u(t)$ and the simulated value $u(t)$ is correct and identically to the Γ_t as the real execution of the "Private Key Distribution". Similarly, the simulated value $r_t P$ can be generated correctly.

Then we prove "Signing" is simulatable. Given the system parameters *params* the identity ID, the message m, the corresponding signature $\sigma = (T, v, rP, a, b, S)$, $t-1$ private key shares (r_i, S_i) and the corresponding verification key shares $(r_i P, e(P, S_i))$. The adversary computes $T_i = r_i H_2(m)$. Let $H(x)$ be a polynomial like function of degree $t-1$ such that $H(0) = T$ and $H(i) = T_i$ for $1 \leq i \leq t-1$. The adversary can compute and broadcast $T(i) = H(i)$ for $t \leq i \leq n$. Similarly, the adversary computes and broadcasts v_i, a_i, b_i, W_i for $t \leq i \leq n$. □

With Theorem 2, Theorem 4 and Lemma 1, we can prove the following:

Theorem 5. *The proposed ID-based threshold signature scheme is secure in the sense of unforgeability.*

6 Concluding Remarks

In this paper, we propose a new ID-based signature scheme without trusted PKG. In our scheme, there is only one PKG who is not assumed to be trusted. We argue that the proposed scheme combines the advantages of both ID-based systems and CA-based systems. We then extend it to be an ID-based threshold signature scheme, which simultaneously overcomes the problem of key escrow and adopts the approach that the private key associated with an identity rather than the master key of PKG is shared. Our scheme is superior to those schemes with distributed PKGs in terms of both the communication and computation complexity.

Acknowledgement

The authors are grateful to Joonsang Baek for his valuable suggestions and comments to this paper.

References

1. J. Baek and Y. Zheng, *Identity-based threshold decryption*, PKC 2004, LNCS 2947, pp.248-261, Springer-Verlag, 2004.
2. J. Baek and Y. Zheng, *Identity-based threshold signature scheme from the bilinear pairings*, IAS'04 track of ITCC'04, pp.124-128, IEEE Computer Society, 2004.
3. P.S.L.M. Barreto, H.Y. Kim, B. Lynn, and M. Scott, *Efficient algorithms for pairings-based cryptosystems*, Advances in Cryptology-Crypto 2002, LNCS 2442, pp.354-368, Springer-Verlag, 2002.
4. D. Boneh and M. Franklin, *Identity-based encryption from the Weil pairings*, Advances in Cryptology-Crypto 2001, LNCS 2139, pp.213-229, Springer-Verlag, 2001.
5. D. Boneh, B. Lynn, and H. Shacham, *Short signatures from the Weil pairings*, Advances in Cryptology-Asiacrypt 2001, LNCS 2248, pp.514-532, Springer-Verlag, 2001.
6. M. Cerccedo, M. Matsumoto and H. Imai, *Efficient and secure multiparty fenera-tion of digital signatrues based on discrete logarithms*, IEEE Trans. Fundamentals., Vol. E76-A, pp.532-545, 1993.
7. J.C. Cha and J.H. Cheon, *An identity-based signature from gap Diffie-Hellman groups*, PKC 2003, LNCS 2567, pp.18-30, Springer-Verlag, 2003.
8. D. Chaum and T.P. Pedersen, *Wallet databases with observers*, Advances in Cryptology-Crypto 1992, LNCS 740, pp.89-105, Springer-Verlag, 1993.
9. L. Chen and C. Kudla, *Identity based authenticated key agreement from pairingss*, Cryptology ePrint Archive, Report 2002/184.
10. X. Chen, F. Zhang, K. Kim, *A new ID-based group signature scheme from bilinear pairings*, Cryptology ePrint Archive, Report 2003/116.
11. G.Frey and H. Rück, *A remark concerning m-divisibility and the discrete logarithm in the divisor class group of curves*, Mathematics of Computation, Vol.62, pp.865-874, 1994.
12. S.D. Galbraith, K. Harrison, and D. Soldera, *Implementing the Tate pairings*, ANTS 2002, LNCS 2369, pp.324-337, Springer-Verlag, 2002.
13. R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin, *Robust threshold DSS signatures*, Advances in Cryptology-Eurocrypt 1996, LNCS 1070, pp.354-371, Springer-Verlag, 1996.
14. C. Gentry and A. Siverberg, *Hierarchical ID-based cryptography*, Advances in Cryptology-Asiacrypt 2002, LNCS 2501, pp.548-566, Springer-Verlag, 2002.
15. C. Gentry, *Certificate-based encryption and the certificate revocation problem*, Advances in Cryptology-Eurocrypt 2003, LNCS 2656, pp.272-293, Springer-Verlag, 2003.
16. M. Girault, *Self-certified public keys*, Advances in Cryptology-Eurocrypt 1991, LNCS 547, pp.490-497, Springer-Verlag, 1991.
17. F. Hess, *Efficient identity based signature schemes based on pairings*, SAC 2002, LNCS 2595, Springer-Verlag, pp.310-324, 2002.
18. B. Libert and J. Quisquater, *Efficient revocation and threshold pairing based cryptosystems*, PODC 2003, ACM Press, pp.163-171, 2003.

19. A. Menezes, T. Okamoto and S. Vanstone, *Reducing elliptic curve logarithms to logarithms in a finite field*, IEEE Transaction on Information Theory, Vol.39, pp.1639-1646, 1993.
20. K.G. Paterson, *ID-based signatures from pairingss on elliptic curves*, Electron. Lett., Vol.38, No.18, pp.1025-1026, 2002.
21. A. Shamir, *How to share a secret*, Communications of the ACM, Vol.22, pp.612-613, 1979.
22. A. Shamir, *Identity-based cryptosystems and signature schemes*, Advances in Cryptology-Crypto 1984, LNCS 196, pp.47-53, Springer-Verlag, 1984.
23. N.P. Smart, *An identity based authenticated key agreement protocol based on the Weil pairings*, Electron. Lett., Vol.38, No.13, pp.630-632, 2002.
24. D. Stinson and R. Strobl, *Provably secure distributed Schnorr signatures and a (t,n) threshold scheme for implicit certificate*, ACISP 2001, LNCS 2119, pp.417-434, Springer-Verlag, 2001.
25. F. Zhang and K. Kim, *ID-based blind signature and ring signature from pairings*, Advances in Cryptology-Asiacrypt 2002, LNCS 2501, pp.533-547, Springer-Verlag, 2002.

Separable Linkable Threshold Ring Signatures

Patrick P. Tsang[1], Victor K. Wei[1], Tony K. Chan[1], Man Ho Au[1], Joseph K. Liu[1], and Duncan S. Wong[2]

[1] Department of Information Engineering,
The Chinese University of Hong Kong,
Shatin, Hong Kong
{pktsang3, kwwei, klchan3, mhau3, ksliu9}@ie.cuhk.edu.hk
[2] Department of Computer Science,
The City University of Hong Kong,
Hong Kong
duncan@cityu.edu.hk

Abstract. A ring signature scheme is a group signature scheme with no group manager to setup a group or revoke a signer. A linkable ring signature, introduced by Liu, et al. [20], additionally allows anyone to determine if two ring signatures are signed by the same group member (a.k.a. they are *linked*). In this paper, we present the first separable linkable ring signature scheme, which also supports an efficient thresholding option. We also present the security model and reduce the security of our scheme to well-known hardness assumptions. In particular, we introduce the security notions of *accusatory linkability* and *non-slanderability* to linkable ring signatures. Our scheme supports "event-oriented" linking. Applications to such linking criterion is discussed.

1 Introduction

Ring Signatures. A ring signature scheme [22] is a group signature scheme [10, 2] with no group manager to setup a group or revoke a signer's identity. Formation of a group is *spontaneous* in a way that diversion group members can be totally unaware of being conscripted to the group. It allows members to *anonymously* sign messages on behalf of their group. Applications include leaking secrets [22] and anonymous identification/authentication for ad hoc groups [6, 13].

Threshold Ring Signatures. Threshold cryptography [12] allows n parties to share the ability to perform a cryptographic operation (e.g., creating a digital signature). Any d parties can perform the operation jointly, whereas it is infeasible for at most $d-1$ to do so. In a (d, n)-threshold ring signature scheme, the generation of a ring signature for a group of n members requires the involvement of at least d members/signers, and yet the signature reveals nothing about the identities of the signers. Schemes in the literature include [6, 19, 24].

Linkable Ring Signatures. The notion of linkable ring signatures was introduced by Liu, et al. [20]. They are ring signatures, but with added linkability: such signatures allow anyone to determine if two signatures are signed by the same

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 384–398, 2004.

group member (in which case the two signatures are said to be "*linked*"). If a user signs only once on behalf of a group, the user still enjoys anonymity similar to that in conventional ring signature schemes. If the user signs multiple times, anyone can tell that these signatures have been generated by the same group member. Applications include leaking sequences of secrets and e-voting [20].

Linkable Threshold Ring Signatures. In [20], a (d, n)-threshold extension to its original linkable ring signature scheme is constructed by concatenating d linkable ring signatures. We note that the construction, though simple and trivial, is not efficient. In particular, the space and time complexities are both $O(dn)$. We give in this paper a construction with time and space complexities both being $O(n)$.

Separability. In [8], Camenisch, et. al. diversified the concept of separability of cryptographic protocols into *perfect separability*, *strong separability* and *weak separability* when describing the users' ability to choose their own cryptographic primitive and system parameters. Separability is of particular importance for ring signature schemes as there is no group manager to coordinate the choice of signature primitive and system parameters for each user. For instance, a ring signature scheme that is only weak separable is not practical at all as it is unlikely to have all group members using the same primitive, system parameters and security parameters. The RSA implementation of [22, 1, 19, 24, 20] are strongly separable while the DL implementation of [1, 19, 20] are only weakly separable.

Event-Oriented Linkability. In [20], one can tell if two ring signatures are linked or not if and only if they are signed on behalf of the same group of members. We call this "*group-oriented*" linkability. We present a new linking criterion that we call "*event-oriented*" linkability in which one can tell if two signatures are linked if and only if they are signed for the same event, despite the fact that they may be signed on behalf of different groups. Event-oriented linkable ring signatures are comparatively more flexible in application. E.g., group settings keep changing frequently in ad-hoc group and most of the ring signatures are signed on behalf of different groups, thus render group-oriented linkability virtually useless. Consider another scenario: The CEOs of a company vote for business decisions. Using linkable ring signatures, they can vote anonymously by ring-signing their votes. However, as the group is fixed throughout the polls, votes among polls can be linked by anybody and information can be derived which means anonymity is in jeopardy. This can be prevented when an event-oriented scheme is used.

1.1 Contributions

Our main contributions include:

- We give the first separable linkable ring signature. It also the first linkable ring signature of the CDS-type ([11]).
- We present a security model for linkable threshold ring signature, and reduce the security of our scheme to well-known hard problem assumptions.
- Our scheme supports bandwidth-efficient threshold signing. The signature size in [20] is $O(dn)$ while ours is $O(n)$, where n is the number of users

and d is the threshold. However, our scheme is *interactive*: insiders interact collaboratively to generate the signature.
- We introduce new security notions to linkable ring signatures: (1) *Non-accusatory linkability* only detects the presence of two "linked" signatures, while *accusatory linkability* additionally outputs the identity of the suspected "double-signer". (2) Strong *non-slanderability* means no coalition can generate signatures accusatorily linked to a victim, while weak *non-slanderability* means that certain coalition may be able to generate signatures accusatorily linked to a victim, but the victim has means to vindicate himself.
- We present a new linking criterion that is "*event-oriented*". Under such linkability, one can tell if two signatures are linked if and only if they are signed for the same event, despite the fact that they may be signed on behalf of different groups.

1.2 Organization

The paper is organized as follows: In Sec. 2, we give some preliminaries. In Sec. 3, we describe the building blocks used in our construction. Then we define our separable linkable threshold signatures in Sec. 4. A construction and its security analysis are presented in Sec. 5. We conclude in Sec. 6.

2 Preliminaries

2.1 Notations and Mathematical Assumptions

Definition 1. *A function $f(\lambda)$ is negligible if for all polynomials $p(\lambda)$, $f(\lambda) < 1/p(\lambda)$ holds for all sufficiently large λ. A function is non-negligible if it is not negligible.*

Definition 2 (Strong RSA Assumption [7, 15, 16]). *Given a safe prime product N, and $z \in QR(N)$, it is infeasible to find $u \in \mathbb{Z}_N^*$ and $e > 1$ such that $u^e = z (\mathrm{mod} N)$, in time polynomial in the size of N.*

Definition 3 (Decisional Diffie-Hellman (DDH) over QR(N) Assumption). *Given a generator g of a cyclic group $QR(N)$, where N is a composite of two primes, the distribution ensembles (g^x, g^y, g^z) and (g^x, g^y, g^{xy}), where $x, y, z \in_R [1, ord(g)]$, are computationally indistinguishable by all PPT algorithm in time polynomial in the size of N.*

2.2 Honest-Verifier Zero-Knowledge (HVZK) Proof of Knowledge Protocols (PoKs)

Every HVZK proof can be turned into a signature scheme by setting the challenge to the hash value of the commitment together with the message to be signed [14]. Such a scheme is proven secure by [21] against existential forgery under adaptively chosen message attack [17] in the random oracle model [4]. Following [9], we call these signature schemes "signatures based on proofs of knowledge", SPK for short. Note that there always exists a corresponding HVZK PoK protocol for every SPK.

3 Basic Building Blocks

In this section, we describe some three-move interactive HVZK PoK protocols that we will use as basic building blocks for our event-oriented linkable threshold ring signature scheme. These protocols all work in finite cyclic groups of quadratic residues modulo safe prime products. For each $i = 1, \ldots, n$, let N_i be a safe-prime product and define the group $G_i \doteq QR(N_i)$ such that its order is of length $\ell_i - 2$ for some $\ell_i \in \mathbb{N}$. Also let g_i, h_i be generators of G_i such that their relative discrete logarithms are not known.

Let $1 < \epsilon \in \mathbb{R}$ be a parameter and let $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q$ be a strong collision-resistant hash function, where q is a κ-bit prime for some security parameter $\kappa \in \mathbb{N}$. Define $\mathcal{N} \doteq \{1, \ldots, n\}$ and $\Gamma_i \doteq \{-2^{\ell_i}q, \ldots, (2^{\ell_i}q)^{\epsilon}\}$.

3.1 Proving the Knowledge of Several Discrete Logarithms

This protocol is a straightforward generalization of the protocol for proving the knowledge of a discrete logarithm over groups of unknown order in [7]. This allows a prover to prove to a verifier the knowledge of n discrete logarithms $x_1, \ldots, x_n \in \mathbb{Z}$ of elements $y_1, \ldots, y_n$ respectively and to the bases $g_1, \ldots, g_n$ respectively. Using the notation in [9], the protocol is denoted by:

$$PK\{(\alpha_1, \ldots, \alpha_n) : \bigwedge_{i=1}^{n} y_i = g_i^{\alpha_i}\}.$$

A prover $\mathcal{P}$ knowing $x_1, \ldots, x_n \in \mathbb{Z}$ such that $y_i = g_i^{x_i}$ for all $i = 1, \ldots, n$ can prove to a verifier $\mathcal{V}$ his/her knowledge as follows.

- (Commit.) $\mathcal{P}$ chooses $r_i \in_R \mathbb{Z}_{(2^{\ell_i}q)^{\epsilon}}$ and computes $t_i \leftarrow g_i^{r_i}$ for all $i = 1, \ldots, n$. $\mathcal{P}$ sends $(t_1, \ldots, t_n)$ to $\mathcal{V}$.
- (Challenge.) $\mathcal{V}$ chooses $c \in_R \mathbb{Z}_q$ and sends it to $\mathcal{P}$.
- (Response.) $\mathcal{P}$ computes, for all $i = 1, \ldots, n$, $s_i \leftarrow r_i - cx_i$ (in $\mathbb{Z}$). $\mathcal{P}$ sends $(s_1, \ldots, s_n)$ to $\mathcal{V}$.

$\mathcal{P}$ verifies by checking, for all $i = 1, \ldots, n$, if $t_i \stackrel{?}{=} g_i^{s_i} y_i^c$.

Theorem 1. *If the Strong RSA assumption holds, the protocol is an HVZK PoK protocol.*

Proof. We omit the proof as it is a straightforward extension of the proof of Lemma 1 in [7]. □

As noted before, the protocol can be turned into a signature scheme by replacing the challenge by the hash of the commitment together with the message M to be signed: $c \leftarrow \mathcal{H}((g_1, y_1)|| \ldots ||(g_n, y_n)||t_1|| \ldots ||t_n||M)$. In this case, the signature is $(c, s_1, \ldots, s_n)$ and the verification becomes:

$$c \stackrel{?}{=} \mathcal{H}((g_1, y_1)|| \ldots ||(g_n, y_n)||g_1^{s_1} y_1^c|| \ldots ||g_n^{s_n} y_n^c||M).$$

Following [9], we denote this signature scheme by:

$$SPK\{(\alpha_1, \ldots, \alpha_n) : \bigwedge_{i=1}^{n} y_i = g_i^{\alpha_i}\}(M).$$

3.2 Proving the Knowledge of d Out of n Equalities of Discrete Logarithms

This protocol is constructed using the techniques described in [11], by combining the PoK for discrete logarithm in [7] and the secret sharing scheme due to Shamir [23]. This allows a prover to prove to a verifier his/her knowledge of some d out of n integers $x_1, \ldots, x_n$, where $x_i = \log_{g_i} y_i = \log_{h_i} v_i$ for all $i = 1, \ldots, n$. The protocol is denoted by:

$$PK\left\{(\alpha_1, \ldots, \alpha_n) : \bigvee_{\mathcal{J} \subseteq \mathcal{N}, |\mathcal{J}|=d} \left(\bigwedge_{i \in \mathcal{J}} y_i = g_i^{\alpha_i} \wedge v_i = h_i^{\alpha_i} \right) \right\}.$$

A prover $\mathcal{P}$ knowing, for all $i \in \mathcal{I}$, $x_i \in \mathbb{Z}$ such that $y_i = g_i^{x_i}$ and $v_i = h_i^{x_i}$, where $\mathcal{I}$ is some subset of $\mathcal{N}$ such that $|\mathcal{I}| = d$, can prove his/her knowledge to a verifier $\mathcal{P}$ as follows.

- (Commit.) $\mathcal{P}$ does the following: For $i \in \mathcal{N} \backslash \mathcal{I}$, select $c_i \overset{R}{\leftarrow} \mathbb{Z}_q$. For all $i \in \mathcal{N}$, select $r_i \overset{R}{\leftarrow} \mathbb{Z}_{(2^{\ell_i} q)^\epsilon}$. Compute

$$t_i \leftarrow \begin{cases} g_i^{r_i}, & i \in \mathcal{I}; \\ g_i^{r_i} y_i^{c_i}, & i \in \mathcal{N} \backslash \mathcal{I}, \end{cases} \text{ and } T_i \leftarrow \begin{cases} h_i^{r_i}, & i \in \mathcal{I}; \\ h_i^{r_i} v_i^{c_i}, & i \in \mathcal{N} \backslash \mathcal{I}. \end{cases}$$

  $\mathcal{P}$ sends $(t_1, \ldots, t_n, T_1, \ldots, T_n)$ to $\mathcal{V}$.
- (Challenge.) $\mathcal{V}$ chooses $c \in_R \mathbb{Z}_q$ and sends it to $\mathcal{P}$.
- (Response.) $\mathcal{P}$ does the following: Compute a polynomial f of degree $\leq n-d$ over $\mathbb{Z}_q$ such that $f(0) = c$ and $f(i) = c_i$ for all $i \in \mathcal{N} \backslash \mathcal{I}$. Compute $c_i \leftarrow f(i)$ for all $i \in \mathcal{I}$. Set

$$s_i \leftarrow \begin{cases} r_i - c_i x_i, & i \in \mathcal{I}; \\ r_i, & i \in \mathcal{N} \backslash \mathcal{I}. \end{cases}$$

  $\mathcal{P}$ sends $(f, s_1, \ldots, s_n)$ to $\mathcal{V}$.

$\mathcal{P}$ verifies by checking if (1) f is a polynomial of degree $\leq n - d$ over $\mathbb{Z}_q$, (2) $f(0) \overset{?}{=} c$, and (3) $t_i \overset{?}{=} y_i^{f(i)} g_i^{s_i}$ and $T_i \overset{?}{=} v_i^{f(i)} h_i^{s_i}$, for all $i = 1, \ldots, n$.

Theorem 2. *If the Strong RSA assumption holds, the protocol is an HVZK PoK protocol.*

Proof. (Proof Sketch) To prove the theorem, it suffices to show that the protocol is correct, sound and statistical HVZK.

- (Correctness.) Straightforward.

– (Soundness.) It suffices to show how a witness can be extracted if given two valid protocol conversations with the same commitment but different challenges. Denoting the two conversation transcripts by $\langle (t_1, \ldots, t_n, T_1, \ldots, T_n), (c), (f, s_1, \ldots, s_n) \rangle$ and $\langle (t_1, \ldots, t_n, T_1, \ldots, T_n), (c'), (f', s'_1, \ldots, s'_n) \rangle$, we have $c \neq c'$ and thus $f(0) \neq f'(0)$. As the degrees of f and f' are at most $n - d$, there are at least d distinct values $\pi_1, \ldots, \pi_d \in \{1, \ldots, n\}$ such that $f(\pi_i) \neq f'(\pi_i)$ for all $i = 1, \ldots, d$. Using arguments in [7], $f(\pi) - f'(\pi)$ divides $s'_\pi - s_\pi$ and therefore an integer $\hat{x}$ such that $y_\pi = g_\pi^{\hat{x}\pi}$ and $v_\pi = h_\pi^{\hat{x}\pi}$ can be computed as: $\hat{x}_\pi \leftarrow (s_\pi - s'_\pi)/(f'(\pi) - f(\pi))$.
Hence a witness $(\hat{x}_{\pi_1}, \ldots, \hat{x}_{\pi_d})$ can be computed from two such transcripts.
– (Statistical HVZK.) To simulate a transcript, a simulator $\mathcal{S}$ first chooses uniformly at random a polynomial f' of degree $n-d$ over $\mathbb{Z}_q$. For all $i = 1, \ldots, n$, $\mathcal{S}$ picks uniformly at random $s'_i \in_R \mathbb{Z}_{(2^{\ell_i} q)^\epsilon}$ and computes $t'_i \leftarrow g_i^{s'_i} y_i^{f'(i)}$. The simulated transcript is: $\langle (t'_1, \ldots, t'_n, T'_1, \ldots, T'_n), (f'(0)), (f', s'_1, \ldots, s'_n) \rangle$.

To prove that the simulation is statistical indistinguishable from real protocol conservations, one should consider, for each $i = 1, \ldots, n$, the probability distribution $P_{S_i}(s_i)$ of the responses of the prover and the probability distribution $P_{S'_i}(s'_i)$ according to which $\mathcal{S}$ chooses s'_i. The statistical distance between the two distributions can be computed to be at most: $2(2^{\ell_i})(q-1)/(2^{\ell_i} q)^\epsilon \leq 2/(2^{\ell_i} q)^{\epsilon - 1}$. The result follows.

□

The protocol can be turned into a signature scheme by replacing the challenge by the hash of the commitment together with the message M to be signed:

$$c \leftarrow \mathcal{H}((g_1, y_1, h_1, v_1)|| \ldots ||(g_n, y_n, h_n, y_n)||t_1|| \ldots ||t_n||T_1|| \ldots ||T_n||M).$$

In this case, the signature is $(f, s_1, \ldots, s_n)$ and step (3) of the verification becomes:

$$c \stackrel{?}{=} \mathcal{H}(\ (g_1, y_1, h_1, v_1)|| \ldots ||(g_n, y_n, h_n, y_n)|| \\ y_1^{c_1} g_1^{s_1}|| \ldots ||y_n^{c_n} g_n^{s_n}||v_1^{c_1} h_1^{s_1}|| \ldots ||v_n^{c_n} h_n^{s_n}||M).$$

We denote this signature scheme by:

$$SPK \left\{ (\alpha_1, \ldots, \alpha_n) : \bigvee_{\mathcal{J} \subseteq \mathcal{N}, |\mathcal{J}| = d} \left(\bigwedge_{i \in \mathcal{J}} y_i = g_i^{\alpha_i} \wedge v_i = h_i^{\alpha_i} \right) \right\} (M).$$

4 Security Model

We give our security model and define relevant security notions.

4.1 Syntax

A *linkable threshold ring signature*, (LTRS) scheme, is a tuple of five algorithms (Key-Gen, Init, Sign, Verify and Link).

- $(sk_i, pk_i) \leftarrow \mathsf{Key\text{-}Gen}(1^{\lambda_i})$ is a PPT algorithm which, on input a security parameter $\lambda_i \in \mathbb{N}$, outputs a private/public key pair (sk_i, pk_i). We denote by $\mathcal{SK}$ and $\mathcal{PK}$ the domains of possible secret keys and public keys, resp. When we say that a public key corresponds to a secret key or vice versa, we mean that the secret/public key pair is an output of Key-Gen.
- $\mathsf{param} \leftarrow \mathsf{Init}(\lambda)$ is a PPT algorithm which, on input a security parameter λ, outputs the publicly known system parameters param which includes λ.
- $\sigma \leftarrow \mathsf{Sign}(\lambda, \mathsf{param}, e, n, d, \mathcal{Y}, \mathcal{X}, M)$ is a PPT algorithm which, on input the system parameters param, an event-id $e \in \mathcal{EID}$ (where $\mathcal{EID}$ is the domain of possible event-ids), a group size $n \in \mathbb{N}$ with length polynomial in the security parameter λ, a threshold $d \in \{1, \ldots, n\}$, a set $\mathcal{Y}$ of n public keys in $\mathcal{PK}$, a set $\mathcal{X}$ of d private keys in $\mathcal{SK}$ such that their corresponding public keys are in $\mathcal{Y}$, and a message $M \in \mathcal{M}$ (where $\mathcal{M}$ is the domain of possible messages), produces a signature $\sigma \in \Sigma$ (where Σ is the domain of possible signatures).
- $1/0 \leftarrow \mathsf{Verify}(\lambda, \mathsf{param}, e, n, d, \mathcal{Y}, M, \sigma)$ is a deterministic algorithm which, on input the system's parameters param, an event-id $e \in \mathcal{EID}$, a group size $n \in \mathbb{N}$ with length polynomial in the security parameter λ, a threshold $d \in \{1, \ldots, n\}$, a set $\mathcal{Y}$ of n public keys in $\mathcal{PK}$, a message $M \in \mathcal{M}$ and a signature $\sigma \in \Sigma$, returns 1 or 0 for accept or reject, resp.
- $1/0 \leftarrow \mathsf{Link}(\mathsf{param}, e, (n_1, d_1, \mathcal{Y}_1, M_1, \sigma_1), (n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2))$ is an algorithm which, on input the system's parameters param, an event-id $e \in \mathcal{EID}$, two group sizes $n_1, n_2 \in \mathbb{N}$ with lengths polynomial in the security parameter λ, two thresholds $d_1 \in \{1, \ldots, n_1\}, d_2 \in \{1, \ldots, n_2\}$, two sets $\mathcal{Y}_1$ and $\mathcal{Y}_2$ of public keys in $\mathcal{PK}$ of sizes n_1 and n_2 resp., two messages $M_1, M_2 \in \mathcal{M}$, and two signatures $\sigma_1, \sigma_2 \in \Sigma$ such that $\mathsf{Verify}(\mathsf{param}, e, n_i, d_i, \mathcal{Y}_i, M_i, \sigma_i)=1$ for $i = 1, 2$, returns 1 or 0 for linked or unlinked, resp. In case of linked, Link additionally outputs the public key $pk^* \in \mathcal{Y}_1 \cap \mathcal{Y}_2$ of the "double-signer".

Remark: Our linkability is different from that of [20]. In [20], Link only outputs linked or unlinked but omits the suspect identity. Our linkability can be called *accusatory linkability* whereas that in [20] can be called *non-accusatory linkability*.

Correctness. LTRS schemes must satisfy:

- (Verification Correctness.) Signatures signed according to specification are accepted during verification.
- (Linking Correctness.) If two signatures are signed for the same event according to specification, then they are linked if and only if the two signatures share a common signer. In the case of linked, the suspect output by Link is exactly the common signer.

4.2 Notions of Security

Security of LTRS schemes has three aspects: unforgeability, anonymity and linkability. Before giving their definition, we consider the following oracles which together model the ability of the adversaries in breaking the security of the schemes.

- $pk_i \leftarrow \mathcal{JO}(\perp)$. The *user joining oracle*, on request, adds a new user to the system. It returns the public key $pk \in \mathcal{PK}$ of the new user.
- $sk_i \leftarrow \mathcal{CO}(pk_i)$. The *corruption oracle*, on input a public key $pk_i \in \mathcal{PK}$ that is a query output of $\mathcal{JO}$, returns the corresponding secret key $sk_i \in \mathcal{SK}$.
- $\sigma \leftarrow \mathcal{SO}(e, n, d, \mathcal{Y}, \mathcal{V}, M)$. The *signing oracle*, on input an event-id $e \in \mathcal{EID}$, a group size $n \in \mathbb{N}$, a threshold $d \in \{1, \ldots, n\}$, a set $\mathcal{Y}$ of n public keys in $\mathcal{PK}$, a set $\mathcal{V} \subseteq \mathcal{Y}$ s.t. $|\mathcal{V}| = d$ and a message $M \in \mathcal{M}$, returns a valid signature σ signed by the set of users whose public keys are in $\mathcal{V}$.

Remark: An alternative approach to specify the $\mathcal{SO}$ is to exclude the signer set $\mathcal{V}$ from the input and have $\mathcal{SO}$ select it according to suitable random distribution. We do not pursue that alternative further.

Unforgeability. Unforgeability for LTRS schemes is defined in the following game between the Simulator $\mathcal{S}$ and the Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$:

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters param.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{S}$ an event-id $e \in \mathcal{EID}$, a group size $n \in \mathbb{N}$, a threshold $d \in \{1, \ldots, n\}$, a set $\mathcal{Y}$ of n public keys in $\mathcal{PK}$, a message $M \in \mathcal{M}$ and a signature $\sigma \in \Sigma$.

$\mathcal{A}$ wins the game if: (1) $\mathsf{Verify}(\mathsf{param}, e, n, d, \mathcal{Y}, M, \sigma) = 1$, (2) all of the public keys in $\mathcal{Y}$ are query outputs of $\mathcal{JO}$, (3) at most $(d-1)$ of the public keys in $\mathcal{Y}$ have been input to $\mathcal{CO}$, and (4) σ is not a query output of $\mathcal{SO}$ on any input containing M. We denote by $\mathbf{Adv}_{\mathcal{A}}^{unf}(\lambda)$ the probability of $\mathcal{A}$ winning the game.

Definition 4 (Unforgeability). *An LTRS scheme is unforgeable if for all PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{unf}(\lambda)$ is negligible.*

Linkable Anonymity. Anonymity for LTRS schemes is defined in the following game between the Simulator $\mathcal{S}$ and the Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$:

1. (*Initialization Phase*) $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters param.
2. (*Probe-1 Phase*) $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. (*Gauntlet Phase*) $\mathcal{A}$ gives $\mathcal{S}$ an event-id $e \in \mathcal{EID}$, a group size $n \in \mathbb{N}$, a threshold $d \in \{1, \ldots, n\}$, a message $M \in \mathcal{M}$, and a set $\mathcal{Y}_g$ of n public keys in $\mathcal{PK}$ each of which has been generated by $\mathcal{JO}$ and none of which has been queried to Corruption Oracle $\mathcal{CO}$ or has been included in the insider set $\mathcal{V}$ in any query to Signing Oracle $\mathcal{SO}$.
   $\mathcal{S}$ randomly selects a subset $\mathcal{V}_g \subset \mathcal{Y}_g$, $|\mathcal{V}| = d$, to obtain the d corresponding secret keys by querying $\mathcal{CO}$. $\mathcal{S}$ signs with these secret keys and gives the signature to $\mathcal{A}$.
4. (*Probe-2 Phase*) $\mathcal{A}$ queries the oracles adaptively, except that any member public key of $\mathcal{Y}_g$ cannot be queried to $\mathcal{CO}$ or included in the insider set $\mathcal{V}$ of any query to $\mathcal{SO}$.

5. (*End Game*) $\mathcal{A}$ gives $\mathcal{S}$ a public key $\widetilde{pk} \in \mathcal{Y}$.

$\mathcal{A}$ wins the game if $\widetilde{pk} \in \mathcal{V}_g$. Define

$$\mathbf{Adv}_{\mathcal{A}}^{Anon}(\lambda) = \mathsf{Pr}[\mathcal{A} \text{ wins the game}] - d/n.$$

Definition 5 (Linkable-Anonymity). *An LTRS scheme is linkably-anonymous if for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{Anon}(\lambda)$ is negligible.*

Remark: In the above attacker model, queries to Signing Oracle cannot have any member of $\mathcal{Y}_g$ appearing in the insider set $\mathcal{V}$. The anonymity in [20] is also with respect to this attacker model. We note that our anonymity attacker model is different from those in [3, 5, 18]. We also note that [3], p.623, essentially conjectured that anonymity and linkability cannot coexist in their security model.

Linkability. Linkability for LTRS schemes is defined in the following game between the Simulator $\mathcal{S}$ and the Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$:

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters param.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{S}$ an event-id $e \in \mathcal{EID}$, group sizes $n_1, n_2 \in \mathbb{N}$, thresholds $d_1 \in \{1, \ldots, n_1\}, d_2 \in \{1, \ldots, n_2\}$, sets $\mathcal{Y}_1$ and $\mathcal{Y}_2$ of public keys in $\mathcal{PK}$ of sizes n_1 and n_2 resp., messages $M_1, M_2 \in \mathcal{M}$ and signatures $\sigma_1, \sigma_2 \in \Sigma$.

$\mathcal{A}$ wins the game if (1) all public keys in $\mathcal{Y}_1 \cup \mathcal{Y}_2$ are query outputs of $\mathcal{JO}$, (2) $\mathsf{Verify}(\mathsf{param}, e, n_i, d_i, \mathcal{Y}_i, M_i, \sigma_i) = 1$ for $i = 1, 2$, (3) $\mathcal{CO}$ has been queried at most $(d_1 + d_2 - 1)$ times, and (4) $\mathsf{Link}(\mathsf{param}, e, (n_1, d_1, \mathcal{Y}_1, M_1, \sigma_1), (n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2)) = 0$. We denote by $\mathbf{Adv}_{\mathcal{A}}^{Link}(\lambda)$ the probability of $\mathcal{A}$ winning the game.

Definition 6 (Linkability). *An LTRS scheme is linkable if for all PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{Link}(\lambda)$ is negligible.*

Non-slanderability. Non-slanderability for LTRS schemes is defined in the following game between the Simulator $\mathcal{S}$ and the Adversary $\mathcal{A}$ in which $\mathcal{A}$ is given access to oracles $\mathcal{JO}$, $\mathcal{CO}$ and $\mathcal{SO}$:

1. $\mathcal{S}$ generates and gives $\mathcal{A}$ the system parameters param.
2. $\mathcal{A}$ may query the oracles according to any adaptive strategy.
3. $\mathcal{A}$ gives $\mathcal{S}$ a signature $\sigma_1 \in \Sigma$ and a tuple $(n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2)$.

$\mathcal{A}$ wins the strong game if (1) $\mathsf{Verify}(\mathsf{param}, e, n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2) = 1$, (2) $\mathsf{Link}(\mathsf{param}, e, (n_1, d_1, \mathcal{Y}_1, M_1, \sigma_1), (n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2)) = 1$, and (3) none of the public keys in $\mathcal{V}$ has been input to $\mathcal{CO}$. $\mathcal{A}$ wins the weak game if the following additional constraint holds: (4) the signature σ_1 is a query output of $\mathcal{SO}$. (Let $(e, n_1, d_1, \mathcal{Y}_1, \mathcal{V}, M_1)$ be the associated input tuple), We denote by $\mathbf{Adv}_{\mathcal{A}}^{SNS}(\lambda)$ (resp. $\mathbf{Adv}_{\mathcal{A}}^{WNS}(\lambda)$) the probability of $\mathcal{A}$ winning the strong (resp. weak) game.

Definition 7 (Non-slanderability). *An LTRS scheme is strongly (resp. weakly) non-slanderable if for all PPT adversary $\mathcal{A}$, $\mathbf{Adv}_{\mathcal{A}}^{SNS}(\lambda)$ (resp. $\mathbf{Adv}_{\mathcal{A}}^{WNS}(\lambda)$) is negligible.*

Security. Summarizing we have:

Definition 8 (Security of LTRS Schemes). *An LTRS scheme is secure if it is unforgeable, linkably-anonymous, linkable and weakly non-slanderable.*

5 Our Construction

5.1 An Linkable Threshold Ring Signature Scheme

In this section, we give a concrete construction of an LTRS scheme. We then show that such a construction is secure under the security model defined in the previous section.

- Key-Gen. On input a security parameter ℓ_i, the algorithm randomly picks two distinct primes p_i, q_i of the form $p_i = 2p_i' + 1$ and $q_i = 2q_i' + 1$, where p_i', q_i' are both $((\ell_i - 2)/2)$-bit primes, and sets $N_i \leftarrow p_i q_i$. It then picks a random generator g_i of $QR(N_i)$ and a random $x_i \in_R \mathbb{Z}_{p_i' q_i'}$ and computes $y_i \leftarrow g_i^{x_i}$. It picks a strong collision-resistant hash function $H_i : \{0,1\}^* \rightarrow \{h | \langle h \rangle = QR(N_i)\}$. It sets the public key to $pk_i \leftarrow (\ell_i, N_i, g_i, y_i, H_i)$, and the secret key to $sk_i \leftarrow (p_i, q_i, x_i)$. Finally it outputs (sk_i, pk_i).
- Init. On input security parameters $\ell \in \mathbb{N}$, $1 < \epsilon \in \mathbb{R}$ and $\kappa \in \mathbb{N}$, the algorithm randomly picks a κ-bit prime q and a strong collision-resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$. It outputs the system parameters $\mathsf{param} = (\ell, \epsilon, \kappa, q, H)$.
- Sign. On input the system parameters $\mathsf{param} = (\ell, \epsilon, \kappa, q, H)$, an event-id $e \in \{0,1\}^*$, a group size $n \in \mathbb{N}$, a threshold $d \in \{1, \ldots, n\}$, a public key set $\mathcal{Y} = \{pk_1, \ldots, pk_n\}$, where each $pk_i = (\ell_i, N_i, g_i, y_i, H_i)$ is s.t. $\ell_i \geq \ell$, a private key set $\mathcal{X} = \{sk_{\pi_1}, \ldots, sk_{\pi_d}\}$, where each $sk_{\pi_i} = (p_{\pi_i}, q_{\pi_i}, x_{\pi_i})$ corresponds to $pk_{\pi_i} \in \mathcal{Y}$, and a message $M \in \{0,1\}^*$, Define $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{\pi_1, \ldots, \pi_d\} \subseteq \mathcal{N}$, the algorithm does the following:

  1. For all $i \in \mathcal{N}$, compute $h_{i,e} \leftarrow H_i(\mathsf{param}, pk_i, e)$ and the tags

$$\tilde{y}_{i,e} \leftarrow \begin{cases} h_{i,e}^{x_i}, i \in \mathcal{I}; \\ h_{i,e}^{a_i}, i \in \mathcal{N} \backslash \mathcal{I},\ a_i \overset{R}{\leftarrow} \mathbb{Z}_{\lfloor N_i/4 \rfloor}. \end{cases}$$

  2. Compute a signature $(f, s_1, \ldots, s_n)$ for

$$SPK \left\{ (\alpha_1, \ldots, \alpha_n) : \bigvee_{\mathcal{J} \subseteq \mathcal{N}, |\mathcal{J}| = d} \left(\bigwedge_{i \in \mathcal{J}} y_i = g_i^{\alpha_i} \wedge \tilde{y}_{i,e} = h_{i,e}^{\alpha_i} \right) \right\} (M).$$

  In particular, this requires the knowledge of $x_{\pi_1}, \ldots, x_{\pi_d}$. We will refer to this signature scheme as SPK_1.
  3. Compute a signature $(c, s_1', \ldots, s_n')$ for

$$SPK\left\{(\beta_1,\ldots,\beta_n):\bigwedge_{i=1}^{n}\tilde{y}_{i,e}=h_{i,e}^{\beta_i}\right\}(M).$$

In particular, this requires the knowledge of x_i for all $i \in \mathcal{I}$ and a_i for all $i \in \mathcal{N}\backslash\mathcal{I}$. We will refer to this signature scheme as SPK_2.

4. The signature is

$$\sigma \leftarrow \langle((\tilde{y}_{1,e},\ldots,\tilde{y}_{n,e}),(f,s_1,\ldots,s_n),(c,s'_1,\ldots,s'_n)\rangle.$$

Note that a signature is composed of three parts: the tags, a signature for SPK_1 and a signature for SPK_2.

– Verify. On input a tuple $(\mathsf{param}, e, n, d, \mathcal{Y}, M, \sigma)$, the algorithm parses param into $(\ell, \epsilon, \kappa, q, H)$, $\mathcal{Y}$ into $\{pk_1,\ldots,pk_n\}$, where $pk_i = (\ell_i, N_i, g_i, y_i, H_i)$, and σ into $\langle((\tilde{y}_1,\ldots,\tilde{y}_n),(f,s_1,\ldots,s_n),(c,s'_1,\ldots,s'_n)\rangle$. If any $\ell_i < \ell$, the algorithm returns with 0. Otherwise it does the following:

1. For $i \in \mathcal{N}$, compute $h_{i,e} \leftarrow H_i(\mathsf{param}, pk_i, e)$.
2. Verify if $(f, s_1, \ldots, s_n)$ is a correct signature for SPK_1.
3. Verify if $(c, s'_i, \ldots, s'_n)$ is a correct signature for SPK_2.

– Link. On input a tuple $(\mathsf{param}, e, (n_1, d_1, \mathcal{Y}_1, M_1, \sigma_1), (n_2, d_2, \mathcal{Y}_2, M_2, \sigma_2))$ s.t., for $j = 1, 2$, $\mathsf{Verify}(\mathsf{param}, e, n_j, d_j, \mathcal{Y}_j, M_j, \sigma_j) = 1$, the algorithm first parses, for $j = 1, 2$, $\mathcal{Y}_j$ into $\mathcal{Y}_j = \{pk_1^{(j)},\ldots,pk_{n_j}^{(j)}\}$ and σ_j into

$$\langle((\tilde{y}_{1,e}^{(j)},\ldots,\tilde{y}_{n,e}^{(j)}),(f^{(j)},s_1^{(j)},\ldots,s_n^{(j)}),(c^{(j)},s_1'^{(j)},\ldots,s_n'^{(j)})\rangle.$$

If there exists $\pi_1 \in \{1,\ldots,n_1\}$ and $\pi_2 \in \{1,\ldots,n_2\}$ s.t. $pk_{\pi_1}^{(1)} = pk_{\pi_2}^{(2)}$ and $\tilde{y}_{\pi_1,e}^{(1)} = \tilde{y}_{\pi_2,e}^{(2)}$, it returns 1 and additionally $pk_{\pi_1}^{(1)}$. Otherwise it returns 0.

Correctness. Straightforward.

5.2 Security

We state the security theorems here and provide proof sketches.

Theorem 3 (Unforgeability). *Our construction is unforgeable under the Strong RSA assumption in the random oracle model.*

Proof. (Proof Sketch) Similar to the proof of unforgeability in [19]. Use classification technique and rewinding to produce two signatures with different hash outputs. The soundness from Theorem 2 implies the result.

Simulating Signing Oracle, $\mathcal{SO}$: Upon input (e, n, d, $\mathcal{Y}$, $\mathcal{V}$, M), generate a valid signature as follows: For each $i \in \mathcal{Y}\backslash\mathcal{V}$, randomly generate a_i and compute $\tilde{y}_{i,e} = h_{i,e}^{a_i}$. For each $i \in \mathcal{V}$, randomly generate a_i and backpatch the random oracle to $h_{i,e} = H_i(\mathsf{param}, pk_i, e) = g_i^{a_i}$ and compute $\tilde{y}_{i,e} = y^{a_i}$. Generate $c_0, \cdots, c_n$ such that they interpolate a polynomial f with degree $\leq n - d$ and $f(i) = c_i$ for $0 \leq i \leq n$. For each i, simulate the corresponding 3-move conversation in Step

(2) of Sign with randomly generated responses $s_1, \cdots, s_n$ to produce the commitments. Backpatch the random oracle so that the commitments are hashed to c_0. This completes up to Step (2) in Sign. The rest is easy: Randomly generate challenge c, simulate the SPK in Step (3) of Sign with randomly generate responses $s'_1, \cdots, s'_n$. this Signing Oracle simulation is also used in the proofs of other Theorems below. □

Theorem 4 (Linkable-anonymity). *Our construction is anonymous under the Strong RSA assumption and DDH over QR(N) assumption in the random oracle model.*

Proof. (Proof Sketch) Similar to proof of anonymity in [20]. Let Q_J be an estimate on the number of $\mathcal{JO}$ queries. Denote the Gauntlet DDH Problem as $(\hat{N}, \hat{g}, \hat{g}^{\alpha}, \hat{g}^{\beta}, \hat{g}^{\gamma})$ where $\gamma = \alpha\beta$ with probability 1/2. In the Gauntlet Phase, Simulator $\mathcal{S}$ sets up the witness extraction mechanism as follows: Randomly select $i^* \in \{1, \cdots, Q_J\}$. Return $pk^* \leftarrow (\hat{l}, \hat{N}, \hat{g}, \hat{g}^{\alpha}, \hat{H})$ in the i^*-th $\mathcal{JO}$ query, backpatch Random Oracle $\mathcal{HO}_{i^*}$ to $h_{i,e} = \hat{g}^{\beta}$, generate the Gauntlet signature to $\mathcal{A}$ with $\tilde{y}_{i,e} = \hat{g}^{\gamma}$. If $\mathcal{A}$ returns pk^* as the public key of the accused insider, then $\mathcal{S}$ answers Yes to the DDH question. Otherwise, $\mathcal{S}$ answers Yes or No with equal probability. Denote

$$P_1 = \mathsf{Pr}\{\alpha\beta \not\equiv \gamma \wedge \mathcal{A}\text{'s answer } = pk^*\},\ P_2 = \mathsf{Pr}\{\alpha\beta \equiv \gamma \wedge \mathcal{A}\text{'s answer} = pk^*\},$$
$$P_3 = \mathsf{Pr}\{\alpha\beta \not\equiv \gamma \wedge \mathcal{A}\text{'s answer } \neq pk^*\}\ \ P_4 = \mathsf{Pr}\{\alpha\beta \equiv \gamma \wedge \mathcal{A}\text{'s answer} \neq pk^*\}.$$

Then $\mathsf{Pr}\{\mathcal{A} \text{ wins }\} = P_1 + P_2 \geq d/n + 1/Q(k)$ for some polynomial $Q(k)$. $P_1 = d/(2n)$ because, when DDH Problem is No, all public keys are undistinguishable w.r.t. each other.

$$\begin{aligned}\mathsf{Pr}\{\mathcal{S} \text{ answers DDH correctly}\} &= (P_3 + P_4)/2 + P_2 \\ &= (P_1 + P_2 + P_3 + P_4)/2 + (P_2 - P_1)/2 \\ &\geq \frac{1}{2} + \frac{1}{2}\frac{1}{Q(k)}\end{aligned}$$

□

Theorem 5 (Linkability). *Our construction is linkable under the Strong RSA assumption in the random oracle model.*

Proof. (Proof Sketch) Similar to proof of linkability in [20]. If Adversary can produce two unlinked signatures, then he is rewound twice to produce two sets of witnesses of set-size d_1 and d_2 respectively. If the two sets overlap, then the threshold signatures should have already been linked. If the two sets do not overlap, then we would have obtained a total of $d_1 + d_2$ witnesses while Adversary only corrupted at most $d_1 + d_2 - 1$ witnesses. □

Theorem 6 (Non-slanderability). *Our construction is weakly non-slanderable under the Strong RSA assumption in the random oracle model.*

Proof. (Proof Sketch) The weak non-slanderability is protected by Step (3) of Sign. Given a signature from $\mathcal{SO}$, Adversary does not know the discrete logarithm of any $\tilde{y}_i$, and therefore cannot produce a signature containing some $\tilde{y}_j$ and prove knowledge of logarithm of $\tilde{y}_j$ as in Sign's Step (3). □

Remark: Our scheme does not have strong non-slanderability: User j and User k can agree to use the same $\tilde{y}_i$ to slander User i. User i can vindicate himself by proving that logarithm of his public key y_i does not equal to the logarithm of $\tilde{y}_i$. However, that can be a hassle.

Summarizing, we have:

Theorem 7 (Security). *Our construction is secure under the Strong RSA Assumption and the DDH over QR(N) Assumption in the random oracle model.*

5.3 Discussions

Separability. Users can choose on their own the group of quadratic residues (namely the modulus) as well as the generator of the group. Hence, our construction enjoys strong separability [8]. In fact, we achieved more: users are free to choose their own security parameters (namely the length of modulus). This advantage is vital in applications in ad-hoc environment in which devices with greatly varied computing power are likely to use different security parameters.

Efficiency. In our construction, both the signature size and signing time are of $O(n)$ (n being the group size) but independent of d (the threshold). This is a significant improvement over [20] in which both the size and time complexities are of $O(nd)$. However, our scheme is not non-interactive while [20] is.

Event-IDs. Event-ids should be chosen carefully to according specific applications. We give two examples here. (1) When an event-oriented linkable (threshold) ring signature scheme is used to leak sequences of secrets, the whistle-blower should choose a unique event-id when leaking the first secret and stick to using the same in the sequel. This makes sure that the sequence of secrets cannot be linked to other sequences. (2) When used in electronic voting, it is usually the voting organizer (e.g. the government) who decides on an event-id. Each eligible voter should therefore, before they cast a vote, make sure that the event-id has not been used in any previous voting event, so as to secure the intended privacy.

Linkability in Threshold Ring Signatures. Linkability in threshold ring signatures requires a more precise definition. In particular, there are two possible flavors: two signatures are linked if and only if (1) they are signed by exactly the same set of signers, or (2) they involve a common signer. We call signatures of the former type "*coalition-linkable*" while those of the latter type "*individual-linkable*".

In a coalition-linkable scheme, users are able to sign multiple times without their signatures being linked, as long as they are not collaborating with exactly the same set of signers again. However, in an individual-linkable scheme, a user signing more than once will have the signatures linked, no matter who other collaborating signers are. The scheme we present in this paper falls into the later category.

6 Conclusion

We have given in this paper the first separable linkable ring signature scheme, which also supports an efficient thresholding option. We have also presented the security model and reduce the security of our scheme to well-known hardness assumptions. In particular, we have introduced the security notions of *accusatory linkability* and *non-slanderability* to linkable ring signatures. Applications to event-oriented ring-signing has been discussed.

References

1. M. Abe, M. Ohkubo, and K. Suzuki. 1-out-of-n signatures from a variety of keys. In *ASIACRYPT 2002*, pages 415–432. Springer-Verlag, 2002.
2. G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO 2000*, pages 255–270. Springer-Verlag, 2000.
3. M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: formal definitions, simplified requirements and a construction based on general assumptions. In *EUROCRYPT'03*, volume 2656 of *LNCS*. Springer-Verlag, 2003.
4. M. Bellare and P. Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73. ACM Press, 1993.
5. M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: the case of dynamic groups. Cryptology ePrint Archive, Report 2004/077, 2004. http://eprint.iacr.org/.
6. E. Bresson, J. Stern, and M. Szydlo. Threshold ring signatures and applications to ad-hoc groups. In *Crypto'02*, volume 2442 of *LNCS*, pages 465–480. Springer-Verlag, 2002.
7. J. Camenisch and M. Michels. A group signature scheme based on an RSA-variant. rs RS-98-27, brics, 1998.
8. J. Camenisch and M. Michels. Separability and efficiency for generic group signature schemes. In *Crypto'99*, pages 413–430. Springer-Verlag, 1999.
9. J. Camenisch and M. Stadler. Efficient group signature schemes for large groups (extended abstract). In *CRYPTO'97*, pages 410–424. Springer-Verlag, 1997.
10. D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265. Springer-Verlag, 1991.
11. R. Cramer, I. Damgard, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO'94*, pages 174–187. Springer-Verlag, 1994.
12. Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *CRYPTO '89*, volume 435 of *LNCS*, pages 307–315. Springer-Verlag, 1990.
13. Y. Dodis, A. Kiayias, A. Nicolosi, and V. Shoup. Anonymous identification in ad hoc groups. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 609–626. Springer-Verlag, 2004.
14. A. Fiat and A. Shamir. How to prove yourself: Practical solution to identification and signature problems. In *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer-Verlag, 1987.
15. E. Fujisaki and T. Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO'97*, pages 16–30. Springer-Verlag, 1997.

16. E. Fujisaki and T. Okamoto. A practical and provably secure scheme for publicly verifiable secret sharing and its applications. In *Eurocrypt '98*, volume 1403 of *LNCS*, pages 32–46. Springer-Verlag, 1998.
17. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
18. A. Kiayias and M. Yung. Group signatures: provable security, efficient constructions, and anonymity from trapdoor-holders. Cryptology ePrint Archive, Report 2004/076, 2004. http://eprint.iacr.org/.
19. J. K. Liu, V. K. Wei, and D. S. Wong. A separable threshold ring signature scheme. In *ICISC 2003*, volume 2971 of *LNCS*, pages 12–26. Springer-Verlag, 2003.
20. J. K. Liu, V. K. Wei, and D. S. Wong. Linkable spontaneous anonymous group signature for ad hoc groups (extended abstract). In *ACISP'04*, volume 3108 of *LNCS*, pages 325–335. Springer-Verlag, 2004.
21. D. Pointcheval and J. Stern. Security proofs for signature schemes. In *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer-Verlag, 1996.
22. R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT 2001*, pages 552–565. Springer-Verlag, 2001.
23. A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
24. D. S. Wong, K. Fung, J. K. Liu, and V. K. Wei. On the RS-code construction of ring signature schemes and a threshold setting of RST. In *ICISC 2003*, volume 2971 of *LNCS*, pages 34–46. Springer-Verlag, 2003.

A New Black and White Visual Cryptographic Scheme for General Access Structures

Avishek Adhikari[1], Tridib Kumar Dutta[2], and Bimal Roy[1]

[1] Applied Statistics Unit, Indian Statistical Institute,
203, B T Road, Calcutta 700 035, India
{avishek_r, bimal}@isical.ac.in
[2] SQC Unit, Indian Statistical Institute,
203, B T Road, Calcutta 700 035, India
tridib@isical.ac.in

Abstract. In this paper we introduce a new construction of a black and white visual cryptographic scheme for general access structure. We prove that our scheme gives a strong access structure. We find out the conditions for which our scheme gives less pixel expansion compared to the schemes given in Section 4.2 of [1]. We also propose a modified algorithm giving better result from the pixel expansion's point of view. As a particular case of general access structure, we get (k, n)-VTS with $2 \leq k \leq n$. The (n, n)-VTS obtained from our scheme attains the optimal pixel expansion and the relative contrast. We compare the (k, n)-VTS obtained from the schemes mentioned in Section 4.1 and Section 4.2 of [1] with the (k, n)-VTS obtained from our method and it has been shown that in almost all the cases our pixel expansion is less compared to the other two schemes.

Keywords: secret sharing scheme, visual secret sharing scheme, visual cryptography, visual threshold scheme, general access structure.

1 Introduction

Visual cryptographic scheme, for a set $\mathcal{P}$ of n participants, is a cryptographic paradigm that enables us to split a secret image into n shadow images called *shares*, where each *participant* in $\mathcal{P}$ receives one share. Certain qualified subsets of participants can "visually" recover the secret image with some loss of contrast, but other forbidden sets of participants have no information about the secret image. A "visual recovery" for a set $X \subseteq \mathcal{P}$ consists of photocopying the shares given to the participants in X onto the transparencies, and then stacking them. Since the reconstruction is done by human visual system, no computation is involved during decoding unlike traditional cryptographic schemes where a fair amount of computation is needed to reconstruct the plain text. This cryptographic paradigm was introduced by Naor and Shamir [11]. They analyzed the case of a k out of n Visual Threshold Schemes (VTS) in which the secret image is visible if and only if any k transparencies are stacked together, but if fewer

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 399–413, 2004.

than k transparencies are superimposed it is impossible to decode the original image. The schemes proposed by Naor and Shamir [11] involved black and white images. The concept of Naor and Shamir [11] has been extended in [1] and [2] to general access structure (an access structure is a specification of all qualified and forbidden subsets of participants), where general techniques to construct visual cryptography schemes for any access structure has been proposed.

In this paper we have proposed a new construction of a black and white visual cryptographic scheme for general access structure. We have used the fact that the collection of all the solutions of a system of linear homogeneous equations over the binary field forms a vector space over the base field. We have constructed the basis matrix S^0 whose columns are all the possible solutions of a system of linear homogeneous equations. Another basis matrix S^1 is constructed by considering all solutions of a system of non-homogeneous linear equations.

- We have proved that our scheme gives a strong access structure.
- We have found out the conditions for which our scheme gives lower pixel expansion compared to the schemes given in Section 4.2 of [1].
- We have also proposed a modified algorithm giving better result from the pixel expansion's point of view.
- As a particular case of general access structure, we get (k, n)-VTS with $2 \leq k \leq n$.
- The (n, n)-VTS obtained from our scheme attains the optimal pixel expansion and the relative contrast.
- We compare the (k, n)-VTS obtained from the schemes mentioned in Section 4.1 and Section 4.2 of [1] with the (k, n)-VTS obtained from our method and it has been shown that in almost all the cases our pixel expansion is less compared to the other two schemes. However, compared to [6], the proposed adaptation to (k, n)-VTS has a larger pixel expansion.

2 Preliminaries

2.1 The Model

The model that we describe here is taken nearly verbatim from Blundo, De Santis, and Stinson [3]. Let $\mathcal{P} = \{1, \ldots, n\}$ be a set of elements called *participants*, and let $2^{\mathcal{P}}$ denote the set of all subsets of $\mathcal{P}$. Let Γ_{Qual} and Γ_{Forb} be subsets of $2^{\mathcal{P}}$, where $\Gamma_{Qual} \cap \Gamma_{Forb} = \emptyset$. We will refer to members of Γ_{Qual} as *qualified sets* and the members of Γ_{Forb} as *forbidden sets*. The pair $(\Gamma_{Qual}, \Gamma_{Forb})$ is called the *access structure* of the scheme.

Let us define Γ_0, consisting of all minimal qualified sets, as follows : $\Gamma_0 = \{A \in \Gamma_{Qual} : A' \notin \Gamma_{Qual}, \forall A' \subset A\}$. A participant $P \in \mathcal{P}$ is an *essential* participant if there exists a set $X \subseteq \mathcal{P}$ such that $X \cup \{P\} \in \Gamma_{Qual}$ but $X \notin \Gamma_{Qual}$.

A *monotone increasing access structure* (monotone decreasing access structure) Γ on $\mathcal{P}$ is a subset $\Gamma \subseteq 2^{\mathcal{P}} \setminus \{\phi\}$ ($\Gamma \subseteq 2^{\mathcal{P}}$) such that if $A \in \Gamma$ and $A \subseteq A' \subseteq \mathcal{P}$ ($A' \subseteq A \subseteq \mathcal{P}$), then $A' \in \Gamma$. If Γ_{Qual} is monotone increasing, Γ_{Forb}

is monotone decreasing, and $\Gamma_{Qual} \cup \Gamma_{Forb} = 2^{\mathcal{P}}$, then the access structure is called *strong* or *complete* access structure and Γ_0 is called the *basis* of the access structure.

We assume that the secret image consists of a collection of black and white pixels, each pixel being shared separately. To understand the sharing process consider the case where the secret image consists of just a single black or white pixel. On sharing, this pixel appears in the n shares distributed to the participants. However, in each share the pixel is subdivided into m *subpixels*. This m is called the pixel expansion i.e., the number of pixels, on the transparencies corresponding to the shares (each such pixel is called subpixel), needed to encode one pixel of the original image. It is important to note that the shares are printed on transparencies, and that a "white" subpixel is actually an area where nothing is printed, and therefore left transparent. We assume that the subpixels are sufficiently small and close enough so that the eye averages them to some shade of grey.

In order that the recovered image is clearly discernible, it is important that the grey level of a black pixel be darker than that of a white pixel. Informally, the difference in the grey levels of the two pixel types is called *contrast*. We want the contrast to be as large as possible. Three variables control the perception of black and white regions in the recovered image: a threshold value (t), a relative difference ($\alpha(m)$), and the pixel expansion (m). The *threshold value* is a numeric value that represents a grey level that is perceived by the human eye as the color black. The value $\alpha(m) \cdot m$ is the contrast, which we want to be as large as possible. We require that $\alpha(m) \cdot m \geq 1$ to ensure that black and white areas will be distinguishable. We use "*or*" V to denote the Boolean operation "*or*" of a set of vectors with result V. The *Hamming weight* $w(V)$ is the number of 1's in the Boolean vector V.

2.2 Basis Matrices

To construct a visual cryptographic scheme, it is sufficient to construct the basis matrices corresponding to the black and white pixel. In the following, we formally define what we mean by basis matrices.

Definition 1. *Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be an access structure on a set $\mathcal{P}$ of n participants. A $(\Gamma_{Qual}, \Gamma_{Forb}, m)$-VCS with relative difference $\alpha(m)$ and a set of thresholds $\{t_X\}_{X \in \Gamma_{Qual}}$ is realized using the $n \times m$ basis matrices S^0 and S^1 if the following two conditions hold:*

1. *If $X = \{i_1, i_2, \ldots, i_p\} \in \Gamma_{Qual}$, then the "or" V of the rows $i_1, i_2, \ldots, i_p$ of S^0 satisfies $w(V) \leq t_X - \alpha(m) \cdot m$; whereas, for S^1 it results that $w(V) \geq t_X$.*
2. *If $X = \{i_1, i_2, \ldots, i_p\} \in \Gamma_{Forb}$, the two $p \times m$ matrices obtained by restricting S^0 and S^1 to rows $i_1, i_2, \ldots, i_p$ are equal up to a column permutation.*

2.3 Share Distribution Algorithm

We use the following algorithm to encode the secret image. For each pixel P in the secret image, do the following:

1. Generate a random permutation π of the set $\{1, 2, \ldots, m\}$.
2. If P is a black pixel, then apply π to the columns of S^1; else apply π to the columns of S^0. Call the resulting matrix T.
3. For $1 \leq i \leq n$, row i of T comprises the m subpixels of P in the ith share.

2.4 (n, n)-VTS

Brief description of the construction of the basis matrices [11]:

The basis matrix S^0 is the Boolean matrix whose columns are all the Boolean n-vectors having an even number of 1's; whereas, S^1 is the matrix whose columns are all Boolean n-vectors having an odd number of 1's.

Result 1: ([11]) The above scheme is an n out of n VCS with the parameters $m = 2^{n-1}$ and $\alpha(m) = 1/2^{n-1}$.
Result 2: ([11]) In any (n, n)-VTS, $\alpha(m) \leq 1/2^{n-1}$ and $m \geq 2^{n-1}$.

2.5 General Access Structure

Presently, not many schemes are known about general access structures for VCS. In this section we present two results that are given in [1] for general access structure for black and white visual cryptography.

• Results of VCS Using Cumulative Arrays:
Result 3: [1] Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure, and let Z_M be the family of the maximal forbidden sets in Γ_{Forb}. Then there exists a $(\Gamma_{Qual}, \Gamma_{Forb}, m)$-VCS with $m = 2^{|Z_M|-1}$ and $t_X = m$ for any $X \in \Gamma_{Qual}$.

• Results of VCS from Smaller Schemes:
Result 4: (An adoption of Theorem 4.4 of [1]) Let $(\Gamma'_{Qual}, \Gamma'_{Forb})$ and $(\Gamma''_{Qual}, \Gamma''_{Forb})$ be two access structures on a set of n participants $\mathcal{P}$. If a participant $i \in \mathcal{P}$ is non-essential for $(\Gamma'_{Qual}, \Gamma'_{Forb})$, we assume that $i \in \Gamma'_{Forb}$ and that i does not receive anything as share. Analogously for $(\Gamma''_{Qual}, \Gamma''_{Forb})$. Suppose there exists a $(\Gamma'_{Qual}, \Gamma'_{Forb}, m')$-VCS and a $(\Gamma''_{Qual}, \Gamma''_{Forb}, m'')$-VCS constructed using basis matrices. Then there exists a $(\Gamma'_{Qual} \cup \Gamma''_{Qual}, \Gamma'_{Forb} \cap \Gamma''_{Forb}, m' + m'')$-VCS. If the original access structures are both strong, then so is the resulting access structure.

Note: The construction technique employed in [1] for the above Result does not work for general VCS, i.e., if they are not constructed from basis matrices.

Result 5: [1] Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure having basis Γ_0. Then there exists a $(\Gamma_{Qual}, \Gamma_{Forb}, m)$-VCS where

$$m = \sum_{X \in \Gamma_0} 2^{|X|-1}.$$

In the next Section we will introduce a new black and white visual cryptographic scheme for general access structure.

3 A New Construction for General Access Structure

Let $V = \{1, 2, \ldots, l + k - p\}$, $V_1, V_2 \subseteq V$ with $V = V_1 \cup V_2$ where $|V_1| = l$, $|V_2| = k$, $|V_1 \cap V_2| = p$ such that $0 \leq p < min\{l, k\}$. Let us consider the two systems of linear equations over the binary field given below

$$\left.\begin{array}{l} f_1 = 0 \\ f_2 = 0 \end{array}\right\} \quad \cdots \ (I) \quad and \quad \left.\begin{array}{l} f_1 = 1 \\ f_2 = 1 \end{array}\right\} \quad \cdots (II)$$

where $f_t = \sum_{j \in V_t} v_{tj}$, $t = 1, 2$. The collection of all the solutions of the system of equations (I) forms a vector space of dimension $l + k - p - 2$ over the binary field. So there will be all together $2^{l+k-p-2}$ many solutions of (I). Let us consider a $(l + k - p) \times 2^{l+k-p-2}$ Boolean matrix S^0 whose columns are just all possible solutions of (I).

The system of equations (II) can be written in the form $\mathbf{Av} = \mathbf{b}$. Since the equations of (II) are linearly independent, rank($\mathbf{A}$)=2. Now we know that a system of linear equations $\mathbf{Av} = \mathbf{b}$ is consistent if and only if $\mathbf{A}^T\mathbf{u} = \mathbf{0} \Rightarrow \mathbf{b}^T\mathbf{u} = 0$. Since the rows of A are independent, $\mathbf{A}^T\mathbf{u} = \mathbf{0} \Rightarrow \mathbf{u} = \mathbf{0} \Rightarrow \mathbf{b}^T\mathbf{u} = 0$. Hence, (II) is consistent.

We state the following lemma without proof.

Lemma 1. *If $v = (v_1, v_2, \ldots, v_{l+k-p})$ is any particular solution of the system of equations (II), then all the solutions of (II) can be obtained by just adding v to each solution of (I).*

Now using Lemma 1, we get all possible solutions of (II). Also, the total number of solutions of the system of equations (I) is the same as that of the (II). Let S^1 be a $(l + k - p) \times 2^{l+k-p-2}$ Boolean matrix whose columns are just all possible solutions of (II). Note that if we vary v, we will get different Boolean matrices, but all of them are identical with S^1 up to column permutation. So, without loss of generality, we fix S^0 and S^1.

Now we will prove the following Lemma.

Lemma 2. *Let $\Gamma_0 = \{V_1, V_2\}$ and $\Gamma = \{B \subseteq V : V_1 \subseteq B \text{ or } V_2 \subseteq B\}$. Let 2^V denote the set of all subsets of V, where V_1, V_2 and V are defined above. Let $A \in 2^V \setminus \Gamma$. Then $S^0[A]$ and $S^1[A]$ are identical up to column permutation, where $S^0[A]$ (respectively $S^1[A]$) denotes the restriction of S^0 (respectively S^1) to the rows corresponding to the elements of A.*

Proof : Let us define a set $E_i = \{A \subseteq V : |A| = i\}$, $\forall i = 1, 2, \ldots, l + k - p$. Then $|E_i| = \binom{l+k-p}{i}$.

Let us define two sets as follows $C_1^i = \{A \subseteq V : |A| = i$ and either $V_1 \subseteq A$ *or* $V_2 \subseteq A\}$ and $C_2^i = \{A \subseteq V : |A| = i$ and neither $V_1 \subseteq A$ *nor* $V_2 \subseteq A\}$. Clearly $C_1^i \cup C_2^i = E_i$ and $C_1^i \cap C_2^i = \phi$, $\forall i = 1, 2, \ldots, l + k - p$.
Also $C_1^i \subseteq \Gamma$, $\forall i = min\{l, k\}, \ldots, l + k - p$ and $\cup_{i=min\{l,k\}}^{l+k-p} C_1^i = \Gamma$.

First we consider E_{l+k-p}. Now $|E_{l+k-p}| = 1$. Let $A \in E_{l+k-p}$. Then A contains both V_1 and V_2. So $A \in \Gamma$. Thus $E_{l+k-p} \subseteq \Gamma$.

Now consider $E_{l+k-p-1}$. We find that

$$|C_1^{l+k-p-1}| = \binom{l}{l}\binom{k-p}{k-p-1} + \binom{k}{k}\binom{l-p}{l-p-1} = l+k-2p.$$

So $|C_2^{l+k-p-1}| = \binom{l+k-p}{l+k-p-1} - (l+k-2p) \;=\; p.$

Thus if $p=0$, $C_2^{l+k-p-1} = \phi$.

Let $p \geq 1$. Then there exists exactly p many elements in $C_2^{l+k-p-1}$.

Claim 1 : There exists no $B \in C_2^{l+k-p-1}$ containing all the p elements in V which are common to both V_1 and V_2.

For, if possible let there exist some $B \in C_2^{l+k-p-1}$ which contains all such p elements. Then B must contain $l+k-1-2p$ more elements from V. Now $|V \setminus (V_1 \cap V_2)| = l+k-2p$. So either $V_1 \subseteq B$ or $V_2 \subseteq B$. Then $B \in C_1^{l+k-p-1}$. This contradicts the fact that $C_1^{l+k-p-1} \cap C_2^{l+k-p-1} = \phi$. So there does not exist any $B \in C_2^{l+k-p-1}$ containing all the common p elements of V_1 and V_2. This completes the proof of Claim 1.

Now we consider the p common elements of V_1 and V_2. Without loss of generality, let the common elements be given by $v_{1\ l-p+i} (= v_{2i})$, $i = 1, 2, \ldots, p$. Since $\forall\ B \in C_2^{l+k-p-1}$, $|B| = l+k-p-1$, every $B \in C_2^{l+k-p-1}$ must contain all the elements of V except one of the p common elements of V_1 and V_2. Let $B_1, B_2, \ldots, B_p$ be the p elements of $C_2^{l+k-p-1}$ and let $v_{1\ l-p+i}\ (= v_{2i}) \notin B_i$, $\forall i = 1, 2, \ldots, p$. Then $B_i = V \setminus \{v_{1\ l-p+i}\}$ and $B_i \notin \Gamma$.

Claim 2 : $S^0[B_i]$ and $S^1[B_i]$ are identical up to column permutation, $\forall B_i \in C_2^{l+k-p-1}$, $\forall i = 1, 2, \ldots, p$.

To prove the claim, we see that $v_i = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ with $\leftarrow\ l+k-p\ \rightarrow$ is a solution of the system of equations (II), where 1 is placed at the $l-p+i$ th position, $\forall i = 1, 2, \ldots, p$. So by using Lemma 1, we can construct a Boolean matrix which will be identical with S^0 except for the $(l-p+i)$th row. Also this Boolean matrix is identical up to column permutation with S^1. Thus $S^0[B_i]$ and $S^1[B_i]$ are identical up to column permutation, $\forall B_i \in C_2^{l+k-p-1}$, $i = 1, 2, \ldots, p$. This completes the proof of Claim 2.

Claim 3 : $S^0[B]$ and $S^1[B]$ are identical up to column permutation, $\forall\ B \in C_2^{l+k-p-2}$.

To prove the Claim, we consider $E_{l+k-p-2}$. From the construction of $C_1^{l+k-p-2}$, it is clear that $C_1^{l+k-p-2} \in \Gamma$. Next we consider $C_2^{l+k-p-2}$. Then $C_2^{l+k-p-2} = \{B \subseteq V : |B| = l+k-p-2$ *and neither* $V_1 \subseteq B$ *nor* $V_2 \subseteq B\}$. Then for each $B \in C_2^{l+k-p-2}$, there exists at least one element say $v_{1s} \in V_1$ and there exists at least one element $v_{2t} \in V_2$ such that $v_{1s},\ v_{2t} \notin B$. Let C be the collection of all common elements of V_1 and V_2. Note that C may be empty. Now if one of v_{1s} or v_{2t} is in C, the $V_i = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$, $i = s$ or t is a solution of the system of equations (II) where 1 is at the sth position if $v_{1s} \in C$ or 1 is at

the $(l-p+t)$th position if $v_{2t} \in C$. If both $v_{1s}, v_{2t} \in C$, then both V_s and V_t are solutions of the system of equations (II). Then by using the Lemma 1, we can say that S^0 and S^1 are identical up to column permutation except for the sth or the $(l-p+t)$th row accordingly V_s or V_t is chosen. Now if $v_{1s}, v_{2t} \notin C$, then $\underset{\leftarrow \cdots}{(0,0,\ldots,0,1,0,0,\ldots,0,1,0,\ldots,0)}\underset{l+k-p \qquad \cdots \rightarrow}{}$ is a solution of a system of equations (II), where 1's are at sth position and $(l-p+t)$th position. Then by using the fact of Lemma 1, we can say that S^0 and S^1 are identical up to column permutation except for the sth and $(l-p+t)$th rows, that is $S^0[B]$ and $S^1[B]$ are identical up to column permutation, $\forall\ B \in C_2^{l+k-p-2}$. This completes the proof of Claim 3.

Claim 4 : $S^0[A]$ and $S^1[A]$ are identical up to column permutation for all $A \subseteq V$ such that $1 \leq |A| < l+k-p-2$.

To prove the claim, let us consider any set $A \subseteq V$ such that $1 \leq |A| < l+k-p-2$. Then there are all together three cases. The first two cases are either $V_1 \subseteq A$ or $V_2 \subseteq A$. In those cases $A \in \Gamma$. The third case is neither $V_1 \subseteq A$ nor $V_2 \subseteq A$. In that there must exist some $B \in C_2^{l+k-p-2}$ such that $A \subseteq B$. Since $S^0[B]$ and $S^1[B]$ are identical up to column permutation (by Claim 3), $S^0[A]$ and $S^1[A]$ must also be identical up to column permutation. This completes the proof of Claim 4.

Thus considering the four claims, the lemma follows. □

Now we will prove the following Lemma which will help to prove the main theorem of this paper.

Lemma 3. *Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure on a set $\mathcal{D} = \{1, 2, \ldots, p\}$ of p participants with $\Gamma_0 = \{B_i, B_j\}$ where $p \leq n$, $B_i, B_j \subseteq \mathcal{D}$, $|B_i \cup B_j| = p$, $|B_i| \geq 2$ and $|B_j| \geq 2$. Then there exists a strong visual cryptographic scheme $(\Gamma_{Qual}, \Gamma_{Forb}, m)$ on $\mathcal{D}$ with $m = 2^{|B_i \cup B_j|-2}$ and $t_X = m$, $\forall X \in \Gamma_{Qual}$.*

Proof : Let $B_i = \{i_1, i_2, \ldots, i_l\}$ and $B_j = \{j_1, j_2, \ldots, j_k\}$ where $2 \leq l, k \leq p-1$. For each $i \in \{i_1, i_2, \ldots, i_l\}$ we associate a variable x_i and for each $j \in \{j_1, j_2, \ldots, j_k\}$ we associate a variable x_j. Also for each B_i and B_j we associate functions f_{B_i} and f_{B_j} defined by $f_{B_i} = x_{i_1} + x_{i_2} + \ldots + x_{i_l}$ and $f_{B_j} = x_{j_1} + x_{j_2} + \ldots + x_{j_k}$ respectively. Let us consider two systems of linear equations over the binary field as given below

$$\left.\begin{matrix} f_{B_i} = 0 \\ f_{B_j} = 0 \end{matrix}\right\} \cdots (III) \qquad \text{and} \qquad \left.\begin{matrix} f_{B_i} = 1 \\ f_{B_j} = 1 \end{matrix}\right\} \cdots (IV)$$

Let S^0 and S^1 denote the matrices corresponding to the solutions of (III) and (IV) respectively as defined before. Then S^0 and S^1 are $p \times m$ Boolean matrices where $m = 2^{p-2}$, $p = |B_i \cup B_j|$. Lemma follows from the following claim.

Claim : S^0 and S^1 form basis matrices of a strong VCS $(\Gamma_{Qual}, \Gamma_{Forb}, m)$ for the set $\mathcal{D}$ of p participants with minimal qualified set $\Gamma_0 = \{B_i, B_j\}$, $m = 2^{|B_i \cup B_j|-2}$ and $t_X = m$, $\forall X \in \Gamma_{Qual}$.

Let $B \subseteq \mathcal{D}$ be such that either $B_i \subseteq B$ or $B_j \subseteq B$. To prove the claim, first we will show that B can recover the secret image. We see that $w(S^0_B) \leq 2^{p-2} - 1 =$

$t - m \cdot \alpha(m)$, since the p tuple $(0, 0, \ldots, 0)$ is always a solution of (III). Again, since the number of columns of S^1 is 2^{p-2} and $w(S^1_{B_i}) = 2^{p-2}$, $w(S^1_B) = 2^{p-2}$. Hence B can recover the secret image.

Since the given access structure is a strong access structure, $\Gamma_{Forb} = 2^{\mathcal{P}} \setminus \Gamma_{Qual}$. By using Lemma 2, we have that, for any $A \in \Gamma_{Forb}$, $S^0[A]$ and $S^1[A]$ are identical up to column permutation. Hence the claim. □

Observation 1 : Let $\mathcal{P} = \{1, 2, \ldots, p, p+1, \ldots, n\}$. Then S^0 and S^1 of Lemma 3 can be augmented by a $(n - p) \times m$ null matrix for the non-essential participants $p + 1, p + 2, \ldots, n$, to obtain a strong visual cryptographic scheme $(\Gamma_{Qual}, \Gamma_{Forb}, m)$ on $\mathcal{P}$ with $m = 2^{|B_i \cup B_j|-2}$ and $t_X = m$, $\forall X \in \Gamma_{Qual}$. This construction follows the procedure given in Section 4.2 of [1].
Now we will prove the main theorem of this paper.

Theorem 1. *Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure on a set $\mathcal{P} = \{1, 2, \ldots, n\}$ of n participants with $\Gamma_0 = \{B_1, B_2, \ldots, B_k\}$ where $B_i \subseteq \mathcal{P}, \forall\ i = 1, 2, \ldots, k$. Let σ be a permutation on $\{1, 2, \ldots, n\}$. Then there exists a strong visual cryptographic scheme $(\Gamma_{Qual}, \Gamma_{Forb}, m)$ with $m = M_\sigma$ and $t_X = M_\sigma$, $\forall X \in \Gamma_{Qual}$ where M_σ is given as follows:*

$$M_\sigma = \begin{cases} \sum_{i=1}^{l} 2^{|B_{\sigma(2i-1)} \cup B_{\sigma(2i)}|-2} & if\ k = 2l,\ l \geq 1 \\ \sum_{i=1}^{l} 2^{|B_{\sigma(2i-1)} \cup B_{\sigma(2i)}|-2} + 2^{|B_{\sigma(2l+1)}|-1} & if\ k = 2l+1,\ l \geq 0. \end{cases}$$

Proof : First let $k = 2l$, $l \geq 1$.

Let us define $\Gamma_{0i} = \{B_{\sigma(2i-1)}, B_{\sigma(2i)}\}, \forall\ i = 1, 2, \ldots, l$. Then using Observation 1, we can construct a strong VCS $(\Gamma^i_{Qual}, \Gamma^i_{Forb}, m_i)$ with minimal qualified set Γ_{0i} by constructing two basis matrices S^0_i and S^1_i whose columns are just all possible solutions of the following two systems of linearly independent equations over the binary field respectively concatenated with zero vectors for non-essential participants:

$$\left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 0 \\ f_{B_{\sigma(2i)}} = 0 \end{matrix}\right\} \quad \cdots\ (V) \quad and \quad \left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 1 \\ f_{B_{\sigma(2i)}} = 1 \end{matrix}\right\} \quad \cdots (VI)$$

The strong VCS $(\Gamma^i_{Qual}, \Gamma^i_{Forb}, m_i)$ has the pixel expansion m_i which is given by $m_i = 2^{|B_{\sigma(2i-1)} \cup B_{\sigma(2i)}|-2}$ and $t_{X_i} = m_i$, $\forall X_i \in \Gamma^i_{Qual}$, $\forall i = 1, 2, \ldots l$. By using Result 4, we can construct a strong VCS $(\Gamma_{Qual}, \Gamma_{Form}, m)$ with $m = M_\sigma$ and $t_X = M_\sigma$, $\forall X \in \Gamma_{Qual}$, where M_σ is defined as in the statement of the above theorem.

Let k be odd, that is, let $k = 2l + 1$, $l \geq 0$.

If $l = 0$, then we have $\Gamma_0 = \{B_1\}$. Let $|B_1| = p$. Then we consider two systems of linear equations over the binary field given as follows:

$$f_{B_1} = 0 \quad \cdots\ (VII) \quad and \quad f_{B_1} = 1 \quad \cdots (VIII)$$

Consider two Boolean matrices S^0 and S^1 of order $n \times 2^{p-1}$ whose columns are just all possible solutions of (VII) and (VIII) respectively concatenated with zero vectors for non-essential participants as stated in Observation 1. Now in S^0 there exists exactly one column containing all zeros and in S^1, there does not

exist any column containing all zeros. Also for any $A \subseteq B_1$ with $|A| = p-1$, we have $A = B_1 \setminus \{v\}$ for some $v \in B_1$. Then the p-tuple $(0, 0, \ldots, 0, 1, 0, \ldots, 0)$ is a solution of (VIII), where 1 is at the vth position. Hence by the similar argument as before, $S^0[A]$ and $S^1[A]$ are identical up to column permutation. Hence even if $k = 1$, we get a strong VCS with $\Gamma_0 = \{B_1\}$, $m = 2^{p-1}$ and $t_X = m$, $\forall X \in \Gamma_{Qual}$.

Let $k = 2l+1$, $l \geq 1$. We define $\Gamma_{0i} = \{B_{\sigma(2i-1)}, B_{\sigma(2i)}\}$, $\forall i = 1, 2, \ldots, l$ and $\Gamma_{0\ l+1} = \{B_{\sigma(2l+1)}\}$. As before each of Γ_{0i} forms a strong VCS, $\forall i = 1, 2, \ldots, l+1$ on $\mathcal{P}$. By using Result 4, we can construct a strong VCS $(\Gamma_{Qual}, \Gamma_{Form}, m)$ with $m = M_\sigma$ and $t_X = M_\sigma$, $\forall X \in \Gamma_{Qual}$, where M_σ is defined as in the statement of the above theorem. Hence the result. □

We have computed various steps mentioned in the above theorem through the Example 2 of Appendix.

Note: The value of the pixel expansion m depends on σ. In visual cryptography, one of the aims is to minimize the value of m. So we can choose σ such that M_σ is minimized.

Notations : Let m_{cu} (using cumulative array), m_{Stin} (using smaller schemes) and m_{our} denote respectively the pixel expansions corresponding to the methods given in Subsection 2.5 and Section 3 respectively.

In the next theorem we will find the conditions for which our scheme gives less pixel expansion than the scheme given in Result 5.

Theorem 2. *Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure on a set $\mathcal{P} = \{1, 2, \ldots, b_1 + b_2 - p\}$ of $b_1 + b_2 - p$ participants with $\Gamma_0 = \{B_1, B_2\}$, where $|B_1| = b_1$, $|B_2| = b_2$ and $|B_1 \cap B_2| = p$. Then the strong VCS constructed from the scheme given in Theorem 1 gives less pixel expansion than the scheme given in Result 5 if*

$$p = \begin{cases} b-1, & \text{for } b_1 = b_2 = b \\ b_1 - 1, & \text{for } b_2 > b_1. \end{cases}$$

Proof : According to the method given in Theorem 1, the pixel expansion is $m_{our} = 2^{b_1+b_2-p-2}$. According to the method given in Result 5, the pixel expansion is $m_{Stin} = 2^{b_1-1} + 2^{b_2-1}$.

Let us we consider the following two cases :

Case 1: Let $b_1 = b_2 = b$. Then $2^{2b-p-2} < 2^{b-1} + 2^{b-1}$
if and only if $2^{2b-p-2} < 2^b$
if and only if $p > b-2$.

Since $B_1 \neq B_2$ and $p \geq b_1 - 1$, $p = b-1$. So if we find that $|B_1 \cap B_2| = b-1$, then our method must have less pixel expansion compared to the method given in Result 5.

Case 2: Let $b_1 \neq b_2$. Without loss of generality, let $b_2 > b_1$. Then
$2^{b_1+b_2-p-2} < 2^{b_1-1} + 2^{b_2-1}$
if and only if $2^{b_1+b_2-p-1} < 2^{b_1}(1 + 2^{b_2-b_1})$
if and only if $2^{b_2-p-1} < 1 + 2^{b_2-b_1}$
if and only if $2^{b_2-p-1} \leq 2^{b_2-b_1}$ if and only if $b_2 - p - 1 \leq b_2 - b_1$
if and only if $p \geq b_1 - 1$.

Thus $B_1 \neq B_2$ and $p \geq b_1 - 1$ implies $p = b_1 - 1$. So we find that if $p = b_1 - 1$, our method has less pixel expansion than the method given in Result 5. Hence the result. □

3.1 Algorithm for Less Pixel Expansion

Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure on a set $\mathcal{P} = \{1, 2, \ldots, n\}$ of n participants with $\Gamma_0 = \{B_1, B_2, \ldots, B_k\}$ where $B_i \subseteq \mathcal{P}, \forall\ i = 1, 2, \ldots, k$. Let $|B_i| = b_i$ and $|B_i \cap B_j| = p_{ij}$. We now consider the following algorithm.

- find $\sigma \in S_k$, the symmetric group of order k, that minimizes M_σ.
- find $i \in \{1, 2, \ldots, \lfloor k/2 \rfloor\}$ for which the following condition is satisfied

$$p_{\sigma(2i-1)\sigma(2i)} = \begin{cases} b - 1, & \text{if } b_{\sigma(2i-1)} = b_{\sigma(2i)} = b \\ b_{\sigma(2i-1)} - 1, & \text{if } b_{\sigma(2i)} > b_{\sigma(2i-1)} \end{cases} \cdots (A)$$

  For those i's consider two systems of equations over the binary field,

$$\left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 0 \\ f_{B_{\sigma(2i)}} = 0 \end{matrix}\right\} \text{ and } \left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 1 \\ f_{B_{\sigma(2i)}} = 1 \end{matrix}\right\}$$

  as described in Lemma 3.
- for those i's not satisfying the above condition, instead of making a pair of system of equations

$$\left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 0 \\ f_{B_{\sigma(2i)}} = 0 \end{matrix}\right\} \text{ and } \left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 1 \\ f_{B_{\sigma(2i)}} = 1 \end{matrix}\right\},$$

  we consider the two sets of participants $B_{\sigma(2i-1)}$ and $B_{\sigma(2i)}$ separately and consider two separate system of equations

$$\left.\begin{matrix} f_{B_{\sigma(2i-1)}} = 0 \\ f_{B_{\sigma(2i-1)}} = 1 \end{matrix}\right\} \text{ and } \left.\begin{matrix} f_{B_{\sigma(2i)}} = 0 \\ f_{B_{\sigma(2i)}} = 1 \end{matrix}\right\}$$

  for the two sets of participants $B_{\sigma(2i-1)}$ and $B_{\sigma(2i)}$ respectively.

Note that if there exists some i, $i = 1, 2, \ldots, \lfloor k/2 \rfloor$, which satisfies the above condition (A), the pixel expansion m_{mod} of the modified scheme must satisfy $m_{mod} < m_{Stin}$. Also note that $m_{mod} \leq m_{our}$.

We will illustrate the above algorithm through the following Example.

Example 1.

Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be an access structure on a set of 11 participants with $\Gamma_0 = \{B_1 = \{1, 2, 3\}, B_2 = \{2, 3, 4\}, B_3 = \{5, 6, 7, 8\}, B_4 = \{8, 9, 10, 11\}\}$. Then by taking σ to be the identity permutation on $\{1, 2, 3, 4\}$, we get $m_{our} = 4+32 = 36$. Also $m_{Stin} = 2^2 + 2^2 + 2^3 + 2^3 = 24$. Now we apply our modified algorithm. We find that B_1, B_2 satisfy the condition (A), whereas B_3 and B_4 do not satisfy the condition (A). So we take $\Gamma_{01} = \{B_1, B_2\}$, $\Gamma_{02} = \{B_3\}$ and $\Gamma_{03} = \{B_4\}$. Then $m_{mod} = 4 + 8 + 8 = 20$. Thus we find that in this example m_{mod} is less compared to m_{Stin} and m_{our}.

4 Construction of a (k, n)-VTS from General Access Structure

Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be a strong access structure on a set $\mathcal{P} = \{1, 2, \ldots, n\}$ of n participants. Then we can obtain a (k, n) threshold scheme with $2 \leq k \leq n$ as a particular case of general access structure by taking $\Gamma_0 = \{A \subseteq \mathcal{P} : |A| = k\}$.

In Section 2.5, two methods for the construction of general access structures for visual cryptographic scheme are given. As a particular case of that general access structures we can construct (k, n)-VTS schemes. The first technique is based on cumulative arrays and the (k, n)-VTS constructed from that method has the pixel expansion $m_{ca} = 2\binom{n}{k-1}^{-1}$. Again by using the technique described in Result 5, a strong (k, n)-VTS can also be constructed with pixel expansion $m_{Stin} = \binom{n}{k} \cdot 2^{k-1}$.

We will now give a method to construct a (k, n)-VTS from our proposed general access structure given in Section 3. To construct it we need the following lemma.

Lemma 4. *Let $\mathcal{P} = \{1, 2, \ldots, n\}$. Let us consider all possible k element subsets $B_1, B_2, \ldots, B_l$ of $\mathcal{P}$, where $l = \binom{n}{k}$. If l is even, then there exists a permutation σ on $\{1, 2, \ldots, l\}$ such that $|B_{\sigma(2i-1)} \cap B_{\sigma(2i)}| = k-1$, for all $i = 1, 2, \ldots, l/2$. If l is odd then also there exists a permutation σ on $\{1, 2, \ldots, l\}$ such that $|B_{\sigma(2i-1)} \cap B_{\sigma(2i)}| = k-1$, for all $i = 1, 2, \ldots, \lfloor l/2 \rfloor$.*

Proof : The proof follows directly from [5].

In the next theorem, we will find the relative contrast and the pixel expansion of the scheme obtained as a particular case of our proposed scheme for general access structure.

Theorem 3. *Let $(\Gamma_{Qual}, \Gamma_{Forb})$ be an access structure on a set $\mathcal{P} = \{1, 2, \ldots, n\}$ of n participants with $\Gamma_0 = \{A \subseteq \mathcal{P} : |A| = k\}$, $2 \leq k \leq n$. Then there exists a strong (k, n)-VTS with*

$$m_{our} = \begin{cases} l \cdot 2^{k-2}, & if\ l = \binom{n}{k}\ is\ even \\ (l+1) \cdot 2^{k-2}, & if\ l = \binom{n}{k}\ is\ odd \end{cases}$$

and relative contrast $\alpha(m) = \frac{1}{m_{our}}$.

Proof : Let l be even. Then by Lemma 4 and Theorem 1, it follows that $m_{our} = \frac{l}{2} \cdot 2^{k-1} = l \cdot 2^{k-2}$. Next, let l be odd. Then again by Lemma 4 and Theorem 1, it follows that $m_{our} = \frac{l-1}{2} \cdot 2^{k-1} + 2^{k-1} = (l+1)2^{k-2}$. Lastly, from the construction of the basis matrix of the (k, n)-VTS, it follows that $\alpha(m) = \frac{1}{m_{our}}$. Hence the result.

Observation 2: The (n, n)-VTS, obtained as a particular case of our proposed general access structure, has the pixel expansion 2^{n-1} and relative contrast $1/2^{n-1}$. In [11], Naor and Shamir proved that the optimal pixel expansion and

the optimal relative contrast for any (n, n)-VTS are 2^{n-1} and $1/2^{n-1}$ respectively. Hence our (n, n)-VTS attains the optimal pixel expansion and optimal relative contrast.

4.1 Comparison Between m_{our} and m_{ca} for Threshold Schemes

We see that for $k = n$, $m_{our} = 2^{n-1} = m_{ca}$.

Let $k = 2$ and $n > 2$. Also, let $l = \binom{n}{k}$ be even. Then $m_{our} = \binom{n}{2}$. Also $m_{ca} = 2^{n-1}$. Now $\forall\ n > 2$, $2^{n-1} > \binom{n}{2}$ and hence $m_{ca} > m_{our}\ \forall\ n > 2$ with l even.

Next let l be odd and $k = 2,\ n > 2$. Now if $n = 3$ and $k = 2$, we have $m_{our} = 4$ and $m_{ca} = 4$. So in that case $m_{our} = m_{ca}$. Let $n > 3$. Then $m_{our} = \binom{n}{2} + 1 < 2^{n-1} = m_{ca}$, for all odd $n > 3$.

Let $2 < k < n$. First let l be odd. Then to show $m_{ca} > m_{our}$, it is sufficient to show

$$2\binom{n}{k-1}^{-1} > (l+1)2^{k-2}$$

that is to show $2\binom{n}{k-1}^{-k+1} > l + 1$

that is to show $\binom{n}{k-1} - k + 1 > n$, since $2^n > l + 1,\ \forall\ k$ satisfying $2 < k < n$.

that is to show $\binom{n}{k-1} > n + k - 1$.

Now the minimum value of $\binom{n}{k-1}$ is $\frac{n(n-1)}{2}$ and the maximum value of $n + k - 1$ is $2(n-1)$. Now for $n > 4$, $\frac{n(n-1)}{2} > 2(n-1)$. Thus for $2 < k < n$ and l be odd, $\binom{n}{k-1} > n + k - 1$ and hence $m_{ca} > m_{our}$. The similar result holds for l to be even with $2 < k < n$. Hence

$$m_{ca} \begin{cases} = m_{our}\ for\ n = k \geq 2\ or\ k = 2 \text{ and } n = 3 \\ > m_{our}\ for\ k = 2,\ n > 2 \text{ and } l \text{ be even or } k = 2,\ n > 3 \text{ and } l \text{ be odd} \\ \qquad or\ 2 < k < n. \end{cases}$$

4.2 Comparison Between m_{our} and m_{Stin} for Threshold Schemes

For $n = k \geq 2$, we have $m_{our} = m_{Stin}$. Let $2 \leq k < n$ and l be even. Then $m_{our} = l \cdot 2^{k-2} < l \cdot 2^{k-1} = m_{Stin}$.

Let $2 \leq k < n$ and l be odd. Then
$m_{Stin} = l \cdot 2^{k-1} = l \cdot 2^{k-2} + l \cdot 2^{k-2} > l \cdot 2^{k-2} + 2^{k-2}$ (since $l > 1, \forall k,\ 2 \leq k < n$)$= m_{our}$. So we have

$$m_{Stin} \begin{cases} = m_{our}\ for\ k = n \geq 2 \\ > m_{our}\ for\ 2 \leq k < n. \end{cases}$$

Note : (k, n)-VTS obtained by [6] has a much smaller pixel expansions except when $k = n$, than those obtained in the current adaptation, e.g., for $n = 10$, $k = 9$ the pixel expansion in [6] is 434 while the same is 1280 (present paper). However it may be noted in [6] general access structure was not constructed.

5 Advantages of the Proposed Method

- In Theorem 2, we have proved that under certain conditions our scheme gives less pixel expansion than the scheme given in Section 4.2 of [1].
- As a particular case of general access structure, we can construct a (k, n)-VTS from our proposed scheme. It is shown in Subsections 4.1 and 4.2 that in almost all the cases our (k, n)-VTS gives less pixel expansion compared to the pixel expansion of the (k, n)-VTSs obtained from the schemes given in the Section 2.5.
- The (n, n)-VTS obtained as a particular case of the general access structure proposed by us attains the optimal pixel expansion and relative contrast.

References

1. G. Ateniese, C. Blundo, A. De Santis, and D.R. Stinson, *Visual Cryptography for General Access Structures* Information and Computation, vol.129, pp.86-106, 1996.
2. G. Ateniese, C. Blundo, A. De Santis, and D.R. Stinson, *Constructions and Bounds for Visual Cryptography*, in "23rd International Colloquim on Automata, Languages and Programming" (ICALP '96), F.M. auf der Heide and B. Monien Eds., Vol. 1099 of "Lecture Notes in Computer Science", Springer-Verlag, Berlin, pp. 416-428,1996.
3. C. Blundo, A.De Santis, and D.R. Stinson, *On the Contrast in Visual Cryptography Schemes*, J. Cryptology, vol.12, no.4, pp.261-289, 1999.
4. C. Blundo, A.De Santis and M. Naor, *Visual Cryptography for grey Level Images*, Inf. Process. Lett., Vol. 75, Issue. 6, pp.255-259, 2001.
5. J. R. Bitner, G. Ehrlich, and E. M. Reingold, *Efficient generation of the binary reflected Gray code*, Communications of the ACM, 19, Issue 9, pp. 517-521, 1976.
6. S. Droste, *New Results on Visual Cryptography*, Advance in Cryptography-CRYPTO'96, Lecture Notes in Computer Science, 1109, pp. 401-415, Springer-Verlag, 1996.
7. H. Koga and H. Yamamoto, *Proposal of a Lattice-Based Visual Secret Sharing Scheme for Color and Gray-Scale Images*, IEICE Trans. Fundamentals, Vol. E81-A, No. 6 June 1998.
8. H. Koga and T. Ishihara, *New Constructions of the Lattice-Based Visual Secret Sharing Scheme Using Mixture of Colors*, IEICE Trans. Fundamentals, Vol. E85-A, No. 1 January 2002.
9. H. Koga, M. Iwamoto and H. Yamamoto, *An Analytic Construction of the Visual Secret Scheme for Color Images*, IEICE Trans. Fundamentals, Vol. E84-A, No.1 January 2001.
10. L. A. MacPherson, *Grey Level Visual Cryptography for General Access Structures*, Master's Thesis, University of Waterloo, 2002.

11. M. Naor and A. Shamir, *Visual Cryptography*, Advance in Cryptography, Eurocrypt'94, Lecture Notes in Computer Science 950, pp. 1-12, Springer-Verlag, 1994.
12. V. Rijmen and B. Preneel, *Efficient Colour Visual Encryption or "Shared Colors of Benetton"*, presented at EUROCRYPT '96 Rump Session.

Appendix

Example 2. (To hilight computations of Theorem 1.)

Let us consider a strong access structure on a set $\mathcal{P} = \{1, 2, 3, 4, 5\}$ of 5 participants with $\Gamma_0 = \{B_1 = \{1, 2, 3\}, B_2 = \{1, 2, 4\}, B_3 = \{1, 2, 5\}, B_4 = \{1, 3, 4\}, B_5 = \{1, 3, 5\}, B_6 = \{1, 4, 5\}, B_7 = \{2, 3, 5\}, B_8 = \{2, 4, 5\}, B_9 = \{3, 4, 5\}\}$. Let $\Gamma_{0i} = \{B_{\sigma(2i-1)}, B_{\sigma(2i)}\}$, $\forall i = 1, 2, 3, 4$ and $\Gamma_{05} = \{B_{\sigma(9)}\}$ where $\sigma \in S_9$, the symmetric group of order 9. Let us take σ to be the identity permutation.

For Γ_{01} we consider the following two systems of equations over the binary field:

$$\left.\begin{matrix} v_1 + v_2 + v_3 = 0 \\ v_1 + v_2 + v_4 = 0 \end{matrix}\right\} \cdots (IX) \qquad \text{and} \qquad \left.\begin{matrix} v_1 + v_2 + v_3 = 1 \\ v_1 + v_2 + v_4 = 1 \end{matrix}\right\} \cdots (X).$$

Then the following two Boolean matrices S_1^0 and S_1^1 form the basis matrices for a strong VCS with $\Gamma_{01} = \{B_1, B_2\}$:

$$S_1^0 = \begin{bmatrix} 0\,0\,1\,1 \\ 0\,1\,0\,1 \\ 0\,1\,1\,0 \\ 0\,1\,1\,0 \\ 0\,0\,0\,0 \end{bmatrix} \text{ and } S_1^1 = \begin{bmatrix} 0\,0\,1\,1 \\ 0\,1\,0\,1 \\ 1\,0\,0\,1 \\ 1\,0\,0\,1 \\ 0\,0\,0\,0 \end{bmatrix}. \text{ Similar result hold } \forall i = 2, 3, 4, 5.$$

Then by using the construction as described in Result 4, we get the basis matrices for the strong VCS with Γ_0 as follows

$$S^0 = \begin{bmatrix} \leftarrow S_1^0 \rightarrow \leftarrow & S_2^0 & \rightarrow \leftarrow S_3^0 \rightarrow \leftarrow S_4^0 \rightarrow \leftarrow S_5^0 \rightarrow \\ 0\,0\,1\;\;1 & 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,1\;\;1 & 0\,0\,0\;\;0 & 0\,0\,0\;\;0 \\ 0\,1\,0\;\;1 & 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\;\;0 & 0\,0\,1\;\;1 & 0\,0\,0\;\;0 \\ 0\,1\,1\;\;0 & 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\;\;1 & 0\,1\,0\;\;1 & 0\,0\,1\;\;1 \\ 0\,1\,1\;\;0 & 0\,1\,0\,1\,1\,0\,1\,0\,0\,1\,0\;\;1 & 0\,1\,0\;\;1 & 0\,1\,0\;\;1 \\ 0\,0\,0\;\;0 & 0\,0\,1\,1\,1\,1\,0\,0\,0\,1\,1\;\;0 & 0\,1\,1\;\;0 & 0\,1\,1\;\;0 \end{bmatrix}$$

$$S^1 = \begin{bmatrix} \leftarrow S_1^1 \rightarrow \leftarrow & S_2^1 & \rightarrow \leftarrow S_3^1 \rightarrow \leftarrow S_4^1 \rightarrow \leftarrow S_5^1 \rightarrow \\ 0\,0\,1\;\;1 & 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,1\;\;1 & 0\,0\,0\;\;0 & 0\,0\,0\;\;0 \\ 0\,1\,0\;\;1 & 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,0\;\;0 & 0\,0\,1\;\;1 & 0\,0\,0\;\;0 \\ 1\,0\,0\;\;1 & 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\;\;1 & 0\,1\,0\;\;1 & 0\,0\,1\;\;1 \\ 1\,0\,0\;\;1 & 1\,0\,1\,0\,0\,1\,0\,1\,0\,1\,0\;\;1 & 0\,1\,0\;\;1 & 0\,1\,0\;\;1 \\ 0\,0\,0\;\;0 & 1\,1\,0\,0\,0\,0\,1\,1\,1\,0\,0\;\;1 & 1\,0\,0\;\;1 & 1\,0\,0\;\;1 \end{bmatrix}.$$

Here the pixel expansion is 24.

5.1 Example of a General Access Structure

Here we have implemented a general access structure on a set $\mathcal{P} = \{1, 2, 3\}$ of 3 participants with minimal qualified set $\Gamma_0 = \{\{1, 2\}, \{1, 3\}\}$. Here we have used the basis matrices $S^0 = \begin{bmatrix} 0\,1 \\ 0\,1 \\ 0\,1 \end{bmatrix}$ and $S^1 = \begin{bmatrix} 0\,1 \\ 1\,0 \\ 1\,0 \end{bmatrix}$.

Fig. 1. The Secret Image

Fig. 2. Share 1 : No information about the secret image

Fig. 3. Share 2 : No information about the secret image

Fig. 4. Share 3 : No information about the secret image

Fig. 5. Superimposed image : (Share 1 + Share 2)

Fig. 6. Superimposed image : (Share 1 + Share 3)

Fig. 7. Superimposed image : (Share 2 + Share 3) : No information about the secret image

Fig. 8. Superimposed image : (Share 1 + Share 2 + Share 3)

Identification Algorithms for Sequential Traitor Tracing

Marcel Fernandez and Miguel Soriano*

Department of Telematics Engineering, Universitat Politècnica de Catalunya,
C/ Jordi Girona 1 i 3, Campus Nord, Mod C3, UPC,
08034 Barcelona, Spain
{marcelf, soriano}@mat.upc.es

Abstract. Sequential traitor tracing schemes [10, 11] are generally used to deter piracy in multimedia content delivery scenarios. This is accomplished by embedding the codewords of a code with traceability properties into the content, prior to its delivery. The focus of this paper is on tracing algorithms for sequential traitor tracing schemes, where soft-decision list decoding techniques are applied in order to improve the identification process in the original sequential traitor tracing schemes, where where a brute force approach is used.

1 Introduction

The term *traitor tracing* was first introduced in [3], and broadly speaking consists in a set of mechanisms that protect a digital multimedia content distributor against piracy acts committed by dishonest authorized users. As an application example, we can consider pay-TV systems, where each authorized user is given a set of keys that allows her to decrypt the content. These keys are usually contained in a decoder box. In a collusion attack, a coalition of dishonest users (traitors) get together, and by combining some of their keys, they construct a pirate decoder that tries to hide their identities. If the pirate decoder is found, a traitor tracing scheme allows the distributor to identify at least one of the guilty users, using the keys inside the decoder.

Dynamic traitor tracing [4, 1] provides traceability capabilities in case that the content is *immediately rebroadcasted* after it is decrypted. In [10, 11] it is shown that dynamic traitor tracing is vulnerable against a delayed rebroadcast attack, where the colluders rebroadcast the multimedia content after a given delay. Moreover, the authors discuss a new traceability scheme, that they called *sequential traitor tracing*, that is secure against this later attack and give constructions based on error correcting codes.

Related work in tracing schemes can be found in [2, 9].

* This work has been supported in part by the Spanish Research Council (CICYT) Project TIC2002-00818 (DISQET) and by the IST-FP6 Project 506926 (UBISEC).

A. Canteaut and K. Viswanathan (Eds.): INDOCRYPT 2004, LNCS 3348, pp. 414–429, 2004.

In a sequential traitor tracing scheme, the content is divided into *segments*. Let $W = \{w_1, w_2, \ldots, w_M\}$ be the set of users and let $Q = \{1, 2, \ldots, q\}$ be the mark alphabet. Using Q together with a q-ary watermarking scheme, q versions of each segment are obtained. For each segment, one of the versions of the segment is associated with a particular user according to an $M \times n$ array called a *mark allocation table* T. More precisely, the entry in $T(i, j)$ is the version of segment j that is allocated to user w_i. Therefore, the jth column of the mark allocation table is used by the distributor to assign segment versions to users in the jth time interval.

A coalition of dishonest users $U \subset W$ committs piracy by choosing one of their versions and rebroadcasting it. The distributor (tracer) intercepts the rebroadcasted content, and for each segment extracts the mark and appends it to a sequence $\mathbf{z}$, called the *feedback sequence*, in the following way: $\mathbf{z}$ is initially an empty sequence ($\mathbf{z}_0()$). In segment j the distributor appends the extracted mark z_j to the sequence $\mathbf{z}_{j-1}$ to construct the sequence $\mathbf{z}_j = (z_1, z_2, \ldots, z_j)$. For each segment position j, we define the set

$$V_j(U) = \{z_j | z_j \in T(i, j) : w_i \in U\}$$

A *c-feedback sequence* is a feedback sequence $\mathbf{z} = (z_1, \ldots, z_n)$ that is formed by a coalition U of size at most c. In this case, we have that $z_j \in V_j(U)$ for $j = 1, \ldots, n$. Traitors are identified by looking at the feedback sequence a sufficient number of segments. Whenever a traitor is traced, he is immediately removed from the system. The distributor continues monitorizing the rebroadcasted sequence and disconnects all the subsequent identifiable traitors. After the observation of n segments, all traitors are traced and the algorithm converges.

A sequential traitor tracing scheme, c-TA scheme, consists of a mark allocation table T and a tracing algorithm A, such that:

1. $T = (t_{ij})$ is an $M \times n$ array with entries from Q.
2. A is a mapping $A : Q^* \rightarrow 2^U$ with the property that for any *c-feedback sequence* $\mathbf{z}$, there exists a sequence of integers $0 < d_1 < d_2 < \cdots < d_k \leq n$, $(k \leq c)$ such that

$$A(z_j) = \begin{cases} U_j, \emptyset \neq U_j \subseteq U, \ j = d_1, d_2, \ldots, d_k \text{ and } \bigcup_{l=1}^{k} U_l = U \\ \emptyset, \quad \text{otherwise} \end{cases}$$

The quantity d_k is called the *convergence length* of the tracing process, and is the maximum number of steps needed in the algorithm to identify up to c colluders.

1.1 Our Contribution

In this paper we discuss the construction of tracing algorithms for sequential c-TA schemes. The tracing algorithms in the original work [10, 11] of Safavi-Naini and Wang, consist in a going through the feedback sequence and appropriately incrementing a counter associated with every authorized user, which is basically

a brute force approach. The algorithms we present use of as its underlying routine powerful algebraic soft-decision list decoding mechanisms. Whenever a traitor is found, we evaluate its contribution to the feedback sequence up to that point. The information we extract from this evaluation gives us a hint about who the remaining traitors are, and is introduced back into the the tracing process through the soft-decision techniques.

Moreover, in [11] it is noted, that there is a trade of between convergence length and the alphabet size of the sequential c-TA scheme. In this sense, the recently presented Martirosyan-van Trung codes [15], are optimal for sequential c-TA schemes. We show that our approach also applies to this family of codes.

The paper is organized as follows: In Section 2 we give an overview of the coding theory and traceability concepts used in the rest of the paper. Section 3 presents two algorithms that trace traitors in sequential traitor tracing schemes schemes based in Algebraic-Geometric codes. In Section 4 we show traitors can be traced in the family of recursive Martirosyan-van Trung codes. Finally we give our conclusions in Section 5.

2 Overview of Coding Theory and Traceability Concepts

2.1 Algebraic-Geometric Codes

Let C be a set of vectors of a vector space, $\mathbb{F}_q^n$, then C is called a code. The field $\mathbb{F}_q$ is called the *code alphabet*. The *Hamming distance* $\mathbf{d}(\mathbf{a}, \mathbf{b})$ between two words $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^n$ is the number of positions where $\mathbf{a}$ and $\mathbf{b}$ differ. The *minimum distance* of C, denoted by d, is defined as the smallest distance between two different codewords. A code C with length n, size $|C| = M$, minimum distance d and alphabet $\mathbb{F}_q$ is denoted as an $(n, M, d)_q$ code. For notation convenience, sometimes we will omit the distance and we will refer to C as an $(n, M)_q$ code.

A code C is a *linear code* if it forms a subspace of $\mathbb{F}_q^n$. A code with length n, dimension k and minimum distance d is denoted as a $[n, k, d]$-code.

Now we define the family of Algebraic-geometric codes (AG codes, also known as *geometric Goppa codes*). First we introduce some notation:

- Let $\mathcal{X}$ be an absolutely irreducible curve over $\mathbb{F}_q$ of genus g.
- $P_1, \ldots, P_n$ are $\mathbb{F}_q$-rational points of $\mathcal{X}$.
- $D = P_1 + \cdots + P_n$.
- G is a divisor of $\mathcal{X}$, of degree $\deg(G) < n$, such that $\mathrm{supp}(G) \cap \mathrm{supp}(D) = \emptyset$.
- $\mathcal{L}(G)$ denotes the linear space of G.

Using the Riemann Theorem we have that the dimension $\dim(\mathcal{L}(G))$, of $\mathcal{L}(G)$ over $\mathbb{F}_q$ is finite. Moreover, we also have that $\dim(\mathcal{L}(G)) \geq \deg(G) + 1 - g$.

The divisors D and G define an AG code, $\mathcal{C}_{\mathcal{L}}(D, G)$, as follows:

$$\mathcal{C}_{\mathcal{L}}(D, G) := \{(f(P_1), \ldots, f(P_n)) \mid f \in \mathcal{L}(G)\} \subseteq \mathbb{F}_q^n$$

The parameters of this code are given in the following theorem:

Theorem 1. *[14] $\mathcal{C}_{\mathcal{L}}(D,G)$ is an AG [n,k,d]-code with $d \geq d^* = n - \deg(G)$ and $k = \dim(\mathcal{L}(G)) \geq \deg(G) + 1 - g$. The integer d^* is called the designed distance of the code.*

Decoding AG Codes. The Guruswami-Sudan soft-decision list decoding algorithm. In a communications system the task of the decoder is to estimate the sent codeword from the received word. In *list decoding* [5], the decoder outputs a list of all codewords within distance $e \geq \lfloor \frac{d-1}{2} \rfloor$ of the received word, thus offering a potential way to recover from errors beyond the error correction bound of the code.

In *soft-decision* decoding, the decoding process takes advantage of "side information" generated by the receiver and instead of using the received word symbols, the decoder uses probabilistic reliability information about these received symbols.

In [5, 6] a powerful soft-decision list decoding algorithm for AG codes is given. The algorithm takes as its input a "matrix" of real weights $r_{i,\gamma}$, with $1 \leq i \leq n$ and $\gamma \in \mathbb{F}_q$, and decodes up to the q-ary Johnson bound [6] on list decoding radius, which will we sufficient for our purposes. We do not describe the full algorithm here, but simply state the results we need in the form of a theorem.

Theorem 2. *[6] Let $\mathcal{C}$ be a q-ary AG-code of blocklength n and designed minimum distance $d = n - \deg(G)$, and let $\epsilon > 0$ be an arbitrary constant. For $1 \leq i \leq n$ and $\gamma \in \mathbb{F}_q$, let $r_{i,\gamma}$ be a non-negative real. Then there exists a deterministic algorithm with runtime $\mathrm{poly}(n, q, 1/\epsilon)$ that, when given as input the weights $r_{i,\gamma}$ for $1 \leq i \leq n$ and $\gamma \in \mathbb{F}_q$, finds a list of all codewords $\boldsymbol{c} = (c_1, \ldots, c_n)$ of $\mathcal{C}$ that satisfy*

$$\sum_{i=1}^{n} r_{i,c_i} \geq \sqrt{(n-d)\sum_{i=1}^{n}\sum_{\gamma\in\mathbb{F}_q} r_{i,\gamma}^2} + \epsilon \max r_{i,\gamma} \tag{1}$$

Henceforth we call this algorithm the GS soft-decision algorithm.

2.2 *c*-Traceability Codes

We use the terminology in [13] to describe *traceability codes*.

Let $U \subseteq C$ be any subset of codewords such that $|U| \leq c$. The set of *descendants* of U, denoted $\mathbf{desc}(U)$, is defined as

$$\mathbf{desc}(U) = \{\mathbf{v} \in \mathbb{F}_q^n \ : v_i \in \{a_i : \mathbf{a} \in U\}, 1 \leq i \leq n\}.$$

For a code C and an integer $c \geq 2$, let $U_i \subseteq C$, $i = 1, 2, \ldots, t$ be all the subsets of C such that $|U_i| \leq c$. A code C is a c-IPP (identifiable parent property) code, if for every $\mathbf{v} \in \mathrm{desc}(C)$, we have that

$$\bigcap_{\{i:\mathbf{v}\in\mathrm{desc}(U_i)\}} U_i \neq \emptyset.$$

Now we define an important subclass of IPP codes called c-traceability (c-TA) codes. For $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$ we can define the set of *matching positions* between $\mathbf{x}$ and $\mathbf{y}$ as $\mu(\mathbf{x}, \mathbf{y}) = \{i : x_i = y_i\}$. Let C be a code, then C is a c-TA code if for any $\mathbf{v} \in \mathbf{desc}(U)$ for some U with $|U| \leq c$, there is at least one codeword $\mathbf{u} \in U$ such that $|\mu(\mathbf{v}, \mathbf{u})| > |\mu(\mathbf{v}, \mathbf{w})|$ for any $\mathbf{w} \in C \backslash U$.

Theorem 3 ([13],[7]). *Let C be a $(n, M, d)_q$ error correcting code, if $d > n(1 - 1/c^2)$ then C is a c-traceability code(c-TA).*

Proof. For a proof of the theorem, see for example [12] ∎

2.3 Asymptotically Good c-TA Codes. The Martirosyan-Van Trung Recursive Construction

We now present an "asymptotically good" family of TA codes due to van Trung and Martirosyan [15]. Their construction is based on IPP code concatenation.

A concatenated code is the combination of an *inner* $[n_i, k_i, d_i]$ q_i-ary code, C_{inn}, $(q_i \geq 2)$ with an *outter* $[n_o, k_o, d_o]$ code, C_{out} over the field $\mathbf{F}_{q_i^{k_i}}$. The combination consists in a mapping ϕ, from the elements of $\mathbf{F}_{q_i^{k_i}}$ to the codewords of the inner code C_{inn} $\phi : \mathbf{F}_{q_i^{k_i}} \rightarrow C_{inn}$ that results in a q_i-ary code of length $n_i n_o$ and dimension $k_i k_o$.

Theorem 4. *[15] Let $c \geq 2$ be an integer. Let $n_0 > c^2$ be an integer and let s_0 be an integer with the prime factorization $s_0 = p_1^{e_1} \cdots p_k^{e_k}$ such that $n_0 \leq p_i^{e_i}$ for all $i = 1, \ldots, k$. Then, for all $h \geq 0$ there exists an $(n_h, M_h)_{s_0}$ c-IPP code, where $M_0 = s_0^{\lceil \frac{n_0}{c^2} \rceil}$, $n_0^* = n_0^{\lceil \frac{n_0}{c^2} \rceil}$. and*

$$n_h = n_{h-1} \cdot n_{h-1}^*,\ M_h = M_{h-1}^{\lceil \frac{n_{h-1}^*}{c^2} \rceil},\ n_{h-1}^* = n_{h-2}^{*\lceil \frac{n_{h-2}^*}{c^2} \rceil},$$

The codes in Theorem 4 have the best known asymptotic behavior, which is stated in the following theorem.

Theorem 5. *[15] For any integer $c \geq 2$ and any integer s having the prime factorization $s = p_1^{e_1} \cdots p_k^{e_k}$ with $c^2 < p_i^{e_i}$ for all $i = 1, \ldots, k$, there exists an infinite class of $(n, M)_q$ c-IPP codes for which n is $O((c^2)^{\log^*(M)} \log(M))$.*

As it is discussed in [15], the results in Theorem 4 and Theorem 5 can be applied to TA-codes, if the IPP codes used in the recursion are replaced by TA-codes.

2.4 Constructions of c-TA Schemes Using Error Correcting Codes

In [10, 11] it is shown that sequential traitor tracing schemes can be constructed using q-ary error correcting codes.

Theorem 6. *[10] Let C be an $(n, M, d)_q$ error correcting code. If $d > n - \frac{n}{c^2} + \frac{1}{c}$ then C is a sequential c-traceability scheme which converges in $d = c(n - d + 1)$ steps.*

The following corollary will allow us to use the terms c-TA scheme and c-TA code indistinctly.

Corollary 1. *[10] Mark allocation table of a sequential c-TA scheme is a c-TA code.*

The lemma below, gives a condition to identify traitors that created a given c-feedback sequence.

Lemma 1. *[11] Let C be an $(n, M, d)_q$ code, let $\mathbf{z}_j = (z_1, z_2, \ldots, z_j)$, $j \geq c(n-d)+1$ be a c-feedback sequence produced by $U \subseteq C$, $|U| \leq c$, and let $\mu(\mathbf{z}_j, \mathbf{u})$ denote the number of agreements between $\mathbf{z}_j$ and the codeword $\mathbf{u}$. If $\mu(\mathbf{z}_j, \mathbf{u}) = c(n-d)+1$, then $\mathbf{u} \in U$.*

3 Identification of Traitors in c-TA Schemes Based in AG Codes

We begin the discussion of our tracing algorithms by providing an alternate view of the improvement of the GS algorithm, for the q-ary symmetric erasure channel, given by Koetter and Vardy in [8]. This improvement will turn out to be a key aspect in our tracing proposal.

3.1 Optimal Performance of the GS Algorithm in a q-Ary Symmetric Erasure Channel

The simplest form of soft-decision decoding is called *errors-and-erasures* decoding. An erasure is an indication that the value of a received symbol is in doubt. In this case, when dealing with a q-ary transmission, the decoder has $(q+1)$ output alternatives: the q symbols from $\mathbb{F}_q$, $\gamma_1, \gamma_2, \ldots, \gamma_q$ and $\{*\}$, where the symbol $\{*\}$ denotes an erasure.

In [8], Koetter and Vardy show how to improve the performance of the GS soft-decision algorithm, for the q-ary symmetric erasure channel.

We take $\gamma_1, \gamma_2, \ldots, \gamma_q$ as the ordering of the elements of $\mathbb{F}_q$. Suppose that codeword $\mathbf{u}$ is transmitted and word $\mathbf{v}$ is received. For $1 \leq a \leq n$ and $\gamma_b \in \mathbb{F}_q$, we define:

$$r_{a,b} = \begin{cases} 1/q & \text{if } v_a = \{*\} \\ 1-\delta & \text{if } v_a = \gamma_b \\ \delta/(q-1) & \text{otherwise} \end{cases}$$

If $\mathbf{v}$ contains $(n-m)$ erasures and $(m-l)$ errors, then we have that:

$$\sum_{i=1}^{n} r_{i,c_i} = l(1-\delta) + \frac{(m-l)\delta}{q-1} + \frac{n-m}{q}; \quad \sum_{i=1}^{n} \sum_{\gamma \in \mathbb{F}_q} r_{i,c_i}^2 = m(1-\delta)^2 + \frac{m\delta^2}{q-1} + \frac{n-m}{q}$$

Using (1) we have that

$$\frac{l(1-\delta)+(m-l)\dfrac{\delta}{q-1}+\dfrac{n-m}{q}}{\sqrt{m(1-\delta)^2+m\dfrac{\delta^2}{q-1}+\dfrac{n-m}{q}}} \geq \sqrt{n-d}+\epsilon' \tag{2}$$

where ϵ' is a tolerance parameter as small as we seek.

The left hand of (2), is maximized for

$$\delta = 1-\frac{l}{m}. \tag{3}$$

Using this value in (2), we have that

$$\frac{l^2}{m}+\frac{(m-l)^2}{m(q-1)}+\frac{n-m}{q} \geq (n-d)+\epsilon'' \tag{4}$$

This means that if upon receiving a word $\mathbf{v}$, $(n-m)$ symbols are erased, then for every value of l that satisfies (4) the soft-decoding algorithm will output codeword $\mathbf{u}$. Therefore, the algorithm can handle $(n-m)$ erasures and $(m-l)$ errors.

3.2 Algorithms for Traitor Identification in Sequential c-TA Schemes Based in AG Codes

For a sequential c-TA scheme, the goal of a tracing algorithm is to identify all traitors for a given feedback sequence. If the c-TA scheme is constructed using an error correcting $(n, M, d)_q$ code C, then each authorized user is uniquely associated with a codeword. Therefore we will frequently use the terms user and traitor to denote codewords.

In this section we discuss tracing algorithms for sequential c-TA codes in which the underlying code is an AG code. In this scenario, a given user $u \in C$ is a traitor if the number of agreements between u and the feedback sequence is at least $c(n-d)+1$ as was stated in Lemma 1.

When a user is tagged as a traitor he is immediately disconnected from the system, and thus the number of colluders is reduced. At this point, since there are less than c colluders, the tracing condition can be restated. This situation is expressed, in terms of c-TA codes, as a corollary of Lemma 3 in [11].

Corollary 2. *Let C be a c-traceability $(n, M, d)_q$ code. Let $\mathbf{z}$ be a c-feedback sequence. Suppose that p already identified traitors $(p < c)$ jointly match less than $n-(c-p)(n-d)$ positions of $\mathbf{z}$, then any codeword that agrees with $\mathbf{z}$ in at least $(c-p)(n-d)+1$ of the unmatched positions is also a traitor.*

We first give an intuitive description of the algorithms we present below. Due to the structure of sequential traitor tracing schemes, a feedback sequence

entirely consists in symbols extracted of segments that belong to the colluding traitors. In other words, the symbol in any position of the feedback sequence matches one of the symbols in the segments of one of the traitors, and so the collusion "*covers*" the whole feedback sequence. The tracing process consists in finding users that contribute with a certain number of segments to the feedback sequence. In this case, these users are tagged as traitors and disconnected from the system.

When a traitor is disconnected from the system, his symbol contribution to the feedback sequence are of no use in finding the rest of traitors, and therefore this contribution can be removed from further consideration in the tracing process. A symbol is removed from consideration by erasing it. In other words, upon identifying a traitor, we are able to reduce the number of symbols that we have to deal with, and therefore, we know more about the "look" of the remaining non-identified traitors. The rest of the users that we will consider as traitors can have any symbol in the erased position, but *must* match the feedback sequence in the non-erased symbols.

The following algorithms make use of the function **GS_tracing1**$(\mathbf{z}, p)$, where the argument $\mathbf{z}$ is a c-feedback sequence of length $j \leq n$, and the argument p is an integer that represents the number of already identified traitors that are generating the feedback sequence. The function **GS_tracing1**$(\mathbf{z}, p)$ returns a list containing the codewords of all the provably identifiable traitors that are generating $\mathbf{z}$.

GS_tracing1$(\mathbf{z}, p)$ {
// local variables: l, s, δ

1. For $1 \leq a \leq n$ and $\gamma_b \in \mathbb{F}_q$, initialize the $n \times q$ weights r_{a,γ_b} as follows:

$$r_{a,\gamma_b} := \begin{cases} 1-\delta & \text{if } z_a = \gamma_b \\ 1/q & \text{if } j < a \leq n \text{ or } z_a \text{ erased} \\ \delta/(q-1) & \text{otherwise} \end{cases} \tag{5}$$

2. Set $s :=$ *number of erased positions in* $\mathbf{z}$.
3. Compute the value of l closest to $[(c-p)(n-d)+1]$ that taking $m = n - s$ satisfies (4).
4. Set $\delta = 1 - \frac{l}{n-s}$.
5. Plug in this value of δ in (5) and run the GS soft-decision algorithm.
6. From the output list, **return** all codewords $\mathbf{v}_1, \ldots, \mathbf{v}_t$, that agree with F_j in at least $[(c-p)(n-d)+1]$ of the non-erased positions.
// (Note that t is the number of traitors identified).

}

We observe that The **GS_tracing1**$(\mathbf{z}, p)$ function initializes a $n \times q$ matrix. If the input feedback sequence $\mathbf{z}$ is of length $j < n$, then we extend the feedback sequence $\mathbf{z}$ to a sequence of length n by appending $(n - j)$ erasures. This is the safest thing to do since we don't know anything about the upcoming segments.

The above function is used as a routine in the following algorithm that traces all provably identifiable traitors in a sequential c-TA scheme.

Algorithm 1
Variables:

p	number of identified traitors.
$\mathcal{L}$	List containing the identified traitors.
$\mathbf{z}_j = (z_1, \ldots, z_j)$	c-feedback sequence of length j.
L	Traitors in the output list of the GS algorithm.
s	Number of unprocessed segments.
i	Number of extracted segments.
$\mathbf{u}_m$	mth user's codeword.

Initialization:
$i := 1$, $p := 0$; $\mathcal{L} := \emptyset$; $L := \emptyset$; $s := 0$; $\mathbf{z}_0 = ()$.

Iteration:

// *Wait until tracing is possible*
while $[s \leq (c-p)(n-d)]$ {
  Extract z_i; **Append** z_i to $\mathbf{z}_{i-1}$;
  $s := s + 1$; $i := i + 1$;
}
// *Assertion*: $(c-p)(n-d)+1$ unmatched segments
$L :=$ **GS_tracing1**$(\mathbf{z}_i, p)$;
while $(L == \emptyset)$ {
  Extract z_i; **Append** z_i to $\mathbf{z}_{i-1}$;
  $i := i + 1$;
  $L :=$ **GS_tracing1**$(\mathbf{z}_i, p)$;
}
// *Assertion*: $L \neq \emptyset$
Disconnect all users in L.
$\mathcal{L} := \mathcal{L} \cup L$
for every $(\mathbf{u}_m \in \mathcal{L})$
  Erase the segments of $\mathbf{z}_i$ in $\mu(\mathbf{u}_m, \mathbf{z}_i)$;

Termination:

Set: $s :=$ number of non-erased segments in $\mathbf{z}_i$;
  $p := |\mathcal{L}|$;
if $[(p == c)$ or $(i == n)]$ **quit** // *All traitors have been disconnected.*
else Goto *Iteration*

The most expensive operation in **Algorithm 1** is **GS_tracing1**$(\mathbf{z}, p)$, and depending on the extracted segments (i.e. traitor strategy), it can be executed

several times before it returns a non-empty output. The following algorithm only executes **GS_tracing1**$(\mathbf{z}, p)$, whenever we are sure that a traitor can be found. Next lemma gives a condition about when a parent can be positively identified.

Lemma 2. *Suppose a sequential c-TA scheme based on a c-TA* $(n, M, d)_q$ *code. Let U be a traitor coalition of size c, and suppose that p traitors have already been identified. Then the tracer needs at least* $(c-p)^2(n-d)+1$ *segments from the coalition in order to identify at least one of the* $(c-p)$ *remaining traitors.*

Proof. From [10] (Lemma 3) we have that for a coalition of size c, a traitor can contribute with $c(n-d)$ segments and still be untraceable. This is because two codewords agree in at most $n-d$ symbols. Therefore, for c traitors, the minimum length of a feedback sequence required to identify at least one traitor is $c^2(n-d)+1$. Using the previous reasoning, if p traitors have already been identified, then the coalition is of size $(c-p)$ and can produce $(c-p)^2(n-d)$ segments, before any of the remaining traitors is identifiable. ∎

Now from Lemma 2 it is clear that, in the case that p traitors are already identified, then whenever we have $(c-p)^2(n-d)+1$ unmatched segments in a feedback sequence $\mathbf{z}$, a traitor can be positively identified, and therefore it is in this precise moment that the **GS_tracing1** algorithm will have a non-empty output.

The following algorithm uses the same variables and they are initialized exactly as in **Algorithm 1**, so we omit the declaration and initialization steps.

Algorithm 2
Iteration

while $[s \leq (c-p)^2(n-d)]$ {
  Extract z_i; **Append** z_i to $\mathbf{z}_{i-1}$;
  $s := s+1$; $i := i+1$;
}
// *Assertion*: $(c-p)^2(n-d)+1$ unmatched segments
$L :=$ **GS_tracing1**$(\mathbf{z}_i)$;
// *Assertion*: $L \neq \emptyset$
Disconnect all users in L.
$\mathcal{L} := \mathcal{L} \cup L$
for every $(\mathbf{u}_m \in \mathcal{L})$
  Erase the segments of $\mathbf{z}_i$ in $\mu(\mathbf{u}_m, \mathbf{z}_i)$;

Termination

Set: $s :=$ number of non-erased segments in $\mathbf{z}_i$;
  $p := |\mathcal{L}|$;
if $[(p == c)$ or $(i == n)]$ **quit** // *All traitors have been disconnected.*
else Goto *Iteration*

3.3 Correctness of the Algorithms

To prove the correctness of the algorithm, we only need to show that the function **GS_tracing1**$(\mathbf{z}, p)$ returns a list containing all of the provably identifiable traitors that are generating $\mathbf{z}$.

Since we are using error correcting codes, we relate the construction of a feedback sequence with the transmission of a codeword. In this case, we can say that the "errors in the transmission" are the number of positions in which a traitor and the feedback sequence differ. Using the notation of Section 3.1, we denote by l the number of "correct symbols", by s the number of erased symbols and by m the number of non-erased symbols. In other words, $m = n - s$, and l is the number of non-erased positions in which the feedback sequence and a traitor agree.

Suppose that there are $(c - p)$ unidentified traitors. We first suppose that the number of non-erased symbols is $m \leq (c - p)[(c - p)(n - d) + 1]$. Since in this case, a traitor is a codeword that agrees with the feedback sequence in $l \geq (c - p)(n - d) + 1$ of the non-erased symbols, for every traitor we have that $l^2/m \geq (n - d) + 1/(c - p)$. It follows that (4) is satisfied, and in consequence, all positive parents in this step can be identified.

Now suppose that the number of non-erased symbols is $m > (c - p)[(c - p)(n - d) + 1]$. In this case there exists a traitor, such that, $l \geq m/(c - p)$. For this particular traitor we have that

$$\frac{l^2}{m} \geq \frac{m}{(c-p)^2} > \frac{(c-p)[(c-p)(n-d)+1]}{(c-p)^2} > n - d$$

Again it follows that (4) is satisfied, and therefore this traitor is identified.

Following the above reasoning, it is clear that eventually, all traitors will be traced back.

3.4 Discussion

We want to observe that **Algorithm 2**, is optimal in the sense that in a collusion attack all traitors will contribute to the feedback sequence with the same amount of segments. If this is the case, it is clear that initially we will have to wait for $(c - p)^2(n - d)$ segments before we can identify a traitor. However, we have to keep in mind that traitors are malicious users and therefore some of them can try to cheat the others by contributing with fewer segments. In this the situation, the appropriate algorithm is the one that executes the **GS_tracing1** function at segment intervals between what **Algorithm 1** and **Algorithm 2** do.

4 Identifying Traitors in the Martirosyan-Van Trung Code

As noted in [10, 11] (Theorem 8), for a shorter convergence length a larger code alphabet size is required. In order to overcome this constraint, the authors in [10,

11] suggest the use of code concatenation. In this section we show that when Martirosyan-van Trung codes (that achieve the best known asymptotic behavior) are used in sequential c-TA schemes, tracing algorithms that use soft-decision list decoding techniques can also be applied allowing to trace all provably identifiable traitors.

If we recall the code construction from Theorem 4, we started with codes:

$C_0 : (n_0, M_0)_{s_0}$ c-IPP code with $M_0 = s_0^{\lceil \frac{n_0}{c^2} \rceil}$, and

$C_1^* : (n_0^*, M_1)_{M_0}$ c-IPP code with $n_0^* = n_0^{\lceil \frac{n_0}{c^2} \rceil}$ and $M_1 = M_0^{\lceil \frac{n_0}{c^2} \rceil}$.

Denoting code concatenation with the symbol ||, we have the following sequence of codes:

$$C_1 = C_0||C_1^* \ ; \quad C_2 = C_1||C_2^* \ ; \quad \ldots \ ; \quad C_h = C_{h-1}||C_h^*.$$

where the C_j are the inner codes, the C_k^* are the outter codes, and each C_k^* is an $(n_{k-1}^*, M_k)_{M_{k-1}}$ c-TA code.

Due to the recursive nature of the code, the tracing process will be done in two stages. In the first stage we will need to decode the code C_0, this will be accomplished with the **GS_tracing1** function.

In the second stage of the tracing algorithm we will decode the code C_h by first decoding codes $C_1^*, \ldots, C_{h-1}^*$. To decode code C_i^*, we will use the function **GS_tracing2**. This function has the particularity that instead of accepting a codeword at its input, it accepts a sequence of lists of alphabet symbols. These lists can be processed by using soft-decision decoding techniques, and this is were our advantage comes from, being able to deal with more that one symbol for each position overcomes the limitations of using hard-decision decoding algorithms for tracing in a c-TA code.

The following notation will be useful. We generalize feedback sequences to the case that in each position we can have more than one symbol. That is, for a c-TA code $(n_{i-1}^*, M_i)_{M_{i-1}}$ C_i^*, $\mathbf{Z}$ denotes a feedback sequence, of length at most n_{i-1}^*, of lists of symbols from an alphabet M_{i-1}.

Given a c-TA $(n_{i-1}^*, M_i)_{M_{i-1}}$ C_i^* code, and a feedback sequence of lists of symbols $\mathbf{Z}$, generated by at most $c - p$ codewords $\{\mathbf{u}_1, \ldots, \mathbf{u}_{c-p}\}$, the function **GS_tracing2** returns a list containing all of the provably identifiable codewords (in the sense given by Corollary 2) that can generate the sequence.

The **GS_tracing2** function again uses the GS algorithm as its searching routine. Intuitively the tracing process is as follows: Given a sequence of lists $\mathbf{Z} = (list_1, \ldots, list_i)$, for a given list, the input weights of the GS algorithm are set in the most natural manner. If a symbol is in the list, its weight will be the inverse of the list size and the weight of the symbols not belonging to the list will be zero. With this weight assignment, we execute the GS algorithm. From the output list, we use Corollary 2 to extract all the codewords that provably generated the sequence. Now, we use these extracted codewords to update the lists in $\mathbf{Z}$. If a symbol in $list_i$ is also in the ith position of any of the extracted codewords, then this symbol is removed from the list. Since the size of the lists changes, we set the input weights of the GS algorithm according to the new sizes (if the size of the list is zero then we consider all alphabet symbols equiprobable)

and repeat the whole process until we find all provably identifiable codewords that generated $\mathbf{Z}$.

GS_tracing2$(\mathbf{Z}, p)$:
Input:

- Code C_i^* parameters $(n_{i-1}^*, M_i, M_{i-1})$;
- $\mathbf{Z} = S\_list_1 || \cdots || S\_list_l$, $1 \leq l \leq n_{i-1}^*$ and $S\_list_l = \{s_1, \ldots, s_{|S\_list_l|}\}$, $s_t \in M_{i-1}$.
  // M_{i-1} is the alphabet of the associated code.

Output: A list O_list of codewords of C_i^* that generated $\mathbf{z}$.

1. Set $iter := 0$, $c_{iter} := c - p$.
2. Initialize the $n_{i-1}^* \times M_{i-1}$ weights r_{a,γ_b} for $1 \leq a \leq n_{i-1}^*$, $\forall \gamma_b \in M_{i-1}$ as follows:

$$r_{a,\gamma_b} := \begin{cases} \frac{1}{|S\_list_a|} & \text{if } \exists\ s_t^a = \gamma_b \quad // \ 1 \leq t \leq |S\_list_a| \\ 1/M_{i-1} & \text{if } S\_list_a := \{\emptyset\} \text{ or if } l < a \leq n_{i-1}^* \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

3. With the weights r_{a,γ_b} run the GS algorithm. From the output list take all codewords $\mathbf{u}^j$, such that $u_f^j \in S\_list_f$ for at least $(c_{iter}(n-d)+1)$ values of f, and add them to O_list.
4. Set $iter := iter + 1$, $c_{iter} := c_{iter-1} - j$ and
   $S\_list_f := S\_list_f \ - \ \{s_g^f \mid s_g^f = u_f \text{ for some } \ \mathbf{u} \in O\_list\} \ \ \forall 1 \leq f \leq l$.
5. If $j = 0$ or $c_{iter} = 0$ output O_list and quit, else go to step 2.

We make use of the function **GS_tracing2** the following algorithm that traces all provably identifiable traitors in a c-feedback sequence over a c-traceability Martirosyan-van Trung code.

MvT_tracing$(\mathbf{z}, p)$:
Input:

- Parameters (n_h, M_h, q) of C_h a c-traceability Martirosyan-van Trung code;
- $\mathbf{z}$: c-feedback sequence generated by at most c codewords of C_h.

Output: A list Out_Plist_1 of all codewords (traitors) that generated $\mathbf{z}$.

1. For $cont = 1$ to $\prod_{k=0}^{h-1} n_k^*$.
   - Take symbols $\mathbf{z}_{cont} = (z_{(cont-1)n_0+1}, \ldots, z_{(cont)n_0})$
   - $Out\_PList_{cont}^{(0)}$:=**GS_tracing1**$(\mathbf{z}_{cont}, p)$
2. Set $j := 1$.
3. For $cont' := 1$ to $\prod_{k=j}^{h-1} n_k^*$.

- With the lists $Out\_PList^{(j-1)}_{(cont'-1)n^*_{j-1}+1}, \ldots, Out\_PList^{(j-1)}_{(cont')n^*_{j-1}}$, use the mapping $\phi_{h-j} : M_{j-1} \rightarrow C_{j-1}$ to obtain the lists of symbols $SL^{(j)}_{(cont'-1)n^*_{j-1}+1}, \ldots, SL^{(j)}_{(cont')n^*_{j-1}}$, where $SL_l = \{h_1, \ldots, h_{|SL_l|}\}$, $h_f \in M_{j-1}$.
- $\mathbf{Z} := SL^{(j)}_{(cont'-1)n^*_{j-1}+1} || \cdots || SL^{(j)}_{(cont')n^*_{j-1}}$
- $Out\_PList^{(j)}_{cont'} :=$ **GS_tracing2**$(\mathbf{Z}, p)$

4. Set $j := j + 1$.
5. If $j > h$ output Out_PList_1 (there is only one "surviving" list) else go to Step 3.

Note that since for the code concatenation $C_h = C_{h-1}||C^*_h$, the size of codes C_h and C^*_h is the same, we output the parents as codewords of the code C^*_h.

The **MvT_tracing**$(\mathbf{z}, p)$ routine now can be used instead of the function **GS_tracing1** in **Algorithm 1** and **Algorithm 2**, of Section 3.2, in order to disconnect all traitors in a sequential c-TA scheme.

Note, that the previous algorithms also apply to the case in which the traitors can wait to rebroadcast the content until the broadcast has been completed. Therefore, the algorithms we have presented also apply to c-TA codes when they are used in the traitor tracing schemes in [3].

4.1 Analysis and Correctness of the Algorithm

For c-IPP codes the runtime complexity of the tracing algorithm is in general $O\left(\binom{M}{c}\right)$, whereas for c-traceability codes this complexity is in general $O(M)$, where M is the size of the code. This is where the advantage of c-traceability codes over c-IPP codes comes from. In the van Trung-Martirosyan construction the code C_0 and all C^*_i codes are c-traceability codes, this implies that there exists a tracing algorithm with running time complexity $O(M)$. We achieve the running time $poly(\log M)$ promised in [15], by using the GS algorithm that runs in time polynomial.

To prove the correctness of the algorithm, we only need to show that the function **GS_tracing2**$(\mathbf{Z}, p)$ returns a list containing all of the provably identifiable traitors that are generating $\mathbf{Z}$. We consider the worst case situation, in which all traitors are contributing with the minimum amount of information required for their identification. If there are $c-p$ unidentified traitors, this worst case situation clearly consists in having $(c-p)(n-d)+1$ lists of size $c-p$ and $n-(c-p)(n-d)+1$ empty lists.

We set the $n^*_{i-1} \times M_{i-1}$ weights r_{a,γ_b} according to (2), therefore

$$\sum_{i=1}^{n} \sum_{\gamma \in \mathbb{F}_q} r^2_{i,\gamma} = \frac{(c-p)(n-d)+1}{c-p} + \frac{n-[(c-p)(n-d)+1]}{M_{i-1}}$$

Since a traitor $\mathbf{u}$ contributes at least with $(c-p)(n-d)+1$ symbols, then for $\mathbf{u}$,

$$\sum_{i=1}^{n} r_{i,u_i} = \frac{(c-p)(n-d)+1}{c-p} + \frac{n-[(c-p)(n-d)+1]}{M_{i-1}}$$

We have that

$$\frac{\sum_{i=1}^{n} r_{i,u_i}}{\sqrt{\sum_{i=1}^{n}\sum_{\gamma\in\mathbb{F}_q} r_{i,\gamma}^2}} = \sqrt{(n-d)+\frac{1}{c-p}+\frac{n-[(c-p)(n-d)+1]}{M_{i-1}}}$$

It follows that (1) is satisfied and therefore with the GS algorithm we are able to trace all C_i^* codes.

5 Conclusions

As pointed out in [12] traceability schemes are a worth addition to a system provided its associated algorithms add sufficiently little cost. The focus of this paper is on the efficient identification of traitors in sequential traitor schemes based on c-TA. We show how by carefully using soft-decision decoding techniques we can achieve, in a more efficient manner, the same performance that can be obtained using a brute force approach. Moreover, the algorithms we present when applied the Martirosyan-van Trung code construction achieves the running time complexity promised in [15].

References

1. O. Berkman, M. Parnas, and J. Sgall. Efficient dynamic traitor tracing. *SIAM J. Computing*, 30(6):1802–1828, 2001.
2. D. Boneh and M. Franklin. An efficient public key traitor tracing scheme. *Advances in Cryptology-Crypto 1999, LNCS*, 1666:338–353, 1999.
3. B. Chor, A. Fiat, and M. Naor. Tracing traitors. *Advances in Cryptology-Crypto'94, LNCS*, 839:480–491, 1994.
4. A. Fiat and T. Tassa. Dynamic traitor tracing. *Advances in Cryptology-Crypto 1999, LNCS*, 1666:354–371, 1999.
5. V. Guruswami and M. Sudan. Improved decoding of Reed-Solomon and algebraic-geometry codes. *IEEE Trans. Inform. Theory*, 45(6):1757–1767, 1999.
6. Venkatesan Guruswami. *List Decoding of Error-Correcting Codes*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, September 2001.
7. H. D. L. Hollmann, J. H. van Lint, J. P. Linnartz, and L. M. G. M. Tolhuizen. On codes with the Identifiable Parent Property. *J. Combinatorial Theory*, 82(2):121–133, May 1998.
8. R. Koetter and A. Vardy. Algebraic soft-decision decoding of Reed-Solomon codes. *ISIT'00*, 2000.

9. K. Kurosawa and Y. Desmedt. Optimal traitor tracing and asymetric schemes. *Advances in Cryptology-Eurocrypt 1998, LNCS*, 1438:145–157, 1998.
10. R. Safavi-Naini and Y. Wang. Sequential traitor tracing. *Advances in Cryptology-Crypto 2000, LNCS*, 1880:316–332, 2000.
11. R. Safavi-Naini and Y. Wang. Sequential traitor tracing. *IEEE Trans. Inform. Theory*, 49(5):1319–1326, May 2003.
12. A. Silverberg, J. Staddon, and J. Walker. Efficient traitor tracing algorithms using list decoding. *Advances in Cryptology - ASIACRYPT 2001*, 2248:175 ff., 2001.
13. J. N. Staddon, D. R. Stinson, and R. Wei. Combinatorial properties of frameproof and traceability codes. *IEEE Trans. Inform. Theory*, 47(3):1042–1049, 2001.
14. H. Stichtenoth. *Algebraic Function Fields and Codes*. Berlin: Springer-Verlag, 1993.
15. T. V. Trung and S. Martirosyan. New constructions for ipp codes. *Proc. IEEE International Symposium on Information Theory, ISIT '03*, page 255, 2003.

Author Index

Adhikari, Avishek 399
Aditya, Riza 61
Au, Man Ho 384
Avoine, Gildas 260

Bao, Feng 48, 73
Batten, Lynn Margaret 84
Berger, Thierry 218
Boyd, Colin 1, 17, 61
Braeken, An 120

Chan, Tony K. 384
Chang, Donghoon 328
Chen, Xiaofeng 371

Dalai, Deepak Kumar 92
Dawson, Ed 61
Dutta, Tridib Kumar 399

Feng, Dengguo 73
Fernandez, Marcel 414

Gadiyar, H. Gopalkrishna 305
Galindo, David 245
González Nieto, Juan Manuel 17
Gupta, Kishan Chand 92

Halevi, Shai 315
Harari, Sami 107
Herranz, Javier 356
Hitchcock, Yvonne 17
Hong, Seokhie, 191 328

Kim, Guil 175
Kim, Jongsung 175
Kim, Kwangjo 371
Ko, Youngdai 191
Konidala, Divyan M. 371

Lee, Byoungcheon 61
Lee, Changhoon 191
Lee, Sangjin 175, 191, 328
Lee, Wonil 328
Levy-dit-Vehel, Françoise 275
Lim, Jongin 175
Liu, Joseph K. 384
Loidreau, Pierre 218

Maini, KM Sangeeta 305
Maitra, Subhamoy 92
Martín, Sebastià 245
McAven, Luke 148
McGrew, David A. 343
Monnerat, Jean 260
Montreuil, Audrey 33

Nakahara, Jorge Jr. 162, 206
Nikov, Ventzislav 120
Nikova, Svetla 120

Padma, R. 305
Patarin, Jacques 33
Peng, Kun, 61
Perret, Ludovic 275
Peyrin, Thomas 260
Poinsot, Laurent 107
Preneel, Bart 120

Roy, Bimal 399

Sáez, Germán 356
Safavi-Naini, Reihaneh 148
Sahai, Amit 14
Santana de Freitas, Daniel 206
Sarkar, Palash 230
Sato, Hisayoshi 290
Schepers, Daniel 290
Song, Junghwan 175
Soriano, Miguel 414
Sung, Jaechul 191, 328
Sung, Soohak 328

Takagi, Tsuyoshi 245, 290
Tsang, Patrick P. 384

Viega, John 343
Villar, Jorge L. 245
Vora, Poorvi L. 136

Wang, Guilin 48
Wei, Victor K. 384
Wong, Duncan S. 384
Wu, Hongjun 73

Yung, Moti 148

Zhang, Bin 73
Zhang, Fangguo 371
Zhou, Jianying 48