Cryptography in Radio Frequency Identification and Fair Exchange Protocols

THÈSE N^o 3407 (2005)

PRÉSENTÉE À LA FACULTÉ INFORMATIQUE ET COMMUNICATIONS

Institut de systèmes de communication

SECTION DES SYSTÈMES DE COMMUNICATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Gildas AVOINE

DEA d'intelligence artificielle et algorithmique,
université de Caen Basse-Normandie, France et de nationalité française

devant le jury

Prof. Serge Vaudenay, directeur de thèse

Prof. Touradj Ebrahimi, rapporteur

Prof. André Schiper, rapporteur

Prof. Jacques Stern, rapporteur

Prof. Moti Yung, rapporteur

Lausanne, EPFL
2005

GILDAS AVOINE

Résumé

Cette thèse de doctorat s'intéresse aux protocoles d'échange équitable et aux protocoles d'identification par radiofréquence.

L'échange équitable provient d'un problème de la vie de tous les jours : comment deux personnes peuvent-elles s'échanger des objets (matériels ou immatériels) de manière équitable, c'est-à-dire de telle sorte qu'aucune des deux personnes ne soit lésée dans l'échange ? De manière plus formelle, si Alice et Bob possèdent respectivement les objets m_A et m_B, alors l'échange est équitable si, à la fin du protocole, soit Alice et Bob ont obtenu respectivement m_B et m_A, soit ni Alice ni Bob n'a obtenu l'information attendue, ne serait-ce que partiellement. Assurer l'équité d'un échange est impossible sans ajouter des hypothèses supplémentaires. Nous proposons alors deux approches pour aborder ce problème. La première consiste à adjoindre à chaque personne un *ange gardien*, c'est-à-dire un module de sécurité élaboré par une autorité de confiance et dont le comportement ne peut dévier des règles établies. Dans un tel modèle, l'équité de l'échange peut être assurée avec une probabilité aussi proche de 1 que l'on souhaite, impliquant cependant un coût en terme de complexité. Nous utilisons ensuite des résultats de l'algorithmique distribuée pour généraliser cette approche à n personnes. Enfin, nous proposons une seconde approche qui consiste à ne plus considérer l'échange de manière isolée, mais à le replacer dans son contexte, au centre d'un réseau, où chacune des deux personnes possède certains voisins honnêtes. Dans ce cadre, l'équité peut reposer sur ces voisins, qui ne seront sollicités qu'en cas de conflit durant l'échange.

Nous nous intéressons ensuite à l'identification par radiofréquence (RFID), qui consiste à identifier à distance des objets ou sujets munis d'un transpondeur. L'essor que connaît aujourd'hui cette technologie repose essentiellement sur la volonté de développer des transpondeurs à bas coût et de faible taille, ne disposant par conséquent que de faibles capacités de calcul et de stockage. Pour cette raison, de délicates questions se posent sur le potentiel et les limites de la RFID, notamment en termes de sécurité et de vie privée. Parce que cette problématique est très récente, les travaux présentés dans ce document défrichent avant tout le terrain en posant certains concepts de base. En particulier, nous exhibons et classifions les menaces, nous montrons le lien entre traçabilité et modèle de communication, et nous analysons les protocoles RFID existants. Nous présentons également les problèmes de complexité engendrés par la gestion des clefs. Nous montrons que la solution proposée par Molnar et Wagner présente des faiblesses et suggérons une autre solution reposant sur les compromis temps-mémoire. Enfin, nous poursuivons notre analyse des compromis temps-mémoire en proposant une méthode qui repose sur des *points de contrôle*, qui permettent de détecter de manière probabiliste les fausses alarmes.

GILDAS AVOINE

Abstract

This PhD thesis focuses on fair exchange protocols and radio frequency identification protocols.

Fair exchange stems from a daily life problem: how can two people exchange objects (material or immaterial) fairly, that is, without anyone being hurt in the exchange? More formally, if Alice and Bob each have objects m_A and m_B respectively, then the exchange is fair if, at the end of the protocol, both Alice and Bob have received m_B and m_A respectively, or neither Alice nor Bob have received the expected information, even partially. Ensuring fairness in an exchange is impossible without introducing additional assumptions. Thus, we propose two approaches to overcome this problem. The first consists in attaching to each person, a *guardian angel*, that is, a security module conceived by a trustworthy authority and whose behavior cannot deviate from the established rules. In such a model, the fairness of the exchange can be ensured with a probability as close to 1 as desired, implying however a communication complexity cost. We then use results from the distributed algorithm to generalize this approach for n people. Finally, we propose a second approach that consists in no more considering the exchange in an isolated manner, but to replace it in its context, in the heart of a network, where each person in the pair has a few honest neighbors. In this framework, fairness can lie on these neighbors, who are solicited only in the case of a conflict during the exchange.

We then look into Radio Frequency Identification (RFID), which consists in remotely identifying objects or subjects having a transponder. The great achievements that radio frequency identification has made today, lies essentially on the willingness to develop low cost and small size transponders. Consequently, they have limited computation and storage capabilities. Due to this reason, many questions have been asked regarding RFID's potential and limitations, more precisely in terms of security and privacy. Since this is a recent problem, the works presented in this document first outline completely the framework by introducing certain basic concepts. In particular, we present and classify threats, we show the link between traceability and the communication model, and we analyze existing RFID protocols. We also present the complexity issues due to key management. We show that the solution proposed by Molnar and Wagner has weaknesses and we propose another solution based on time-memory trade-offs. Finally, we continue our time-memory trade-off analysis by proposing a method based on *checkpoints*, which allows detecting false alarms in a probabilistic manner.

À la mémoire de ma mère.

Contents

Remerciements

La cryptographie est vraiment un domaine fascinant. Discipline à la croisée de nombreuses sciences, elle regorge d'anecdotes et d'histoires mystérieuses. À travers les âges, ses techniques et ses objectifs ont évolué, mais son cœur, ce qui la rend si excitante, n'a pas changé : la cryptographie est un jeu constitué d'énigmes à résoudre, d'intrigues à démêler, de défis à relever. Le charme de la cryptographie, c'est aussi le monde qui l'entoure, ce monde de cryptographes, de *cryptopathes* pourrait-on dire, formé de personnes chacune plus singulière que les autres. Prenez n'importe quel sujet de conversation et le cryptopathe le ramènera à sa discipline. Par exemple, installez une nouvelle machine à café électronique et le cryptopathe ne sera satisfait que lorsqu'il aura réussi à la faire boguer. Invitez-le dans un restaurant où l'on peut se resservir à volonté et il tentera une attaque par "rejeu de l'assiette" pour nourrir toute la table. Offrez-lui enfin un ticket de concert électronique et il le disséquera jusqu'au bout afin d'en analyser la sécurité. C'est assurément une curiosité intense et un esprit plus tordu que la moyenne, qui caractérisent le cryptopathe.

Au risque de ne plus boire de café, de ne plus être invité au restaurant et de ne plus écouter Schubert, j'ai choisi de m'investir dans cette discipline et je ne le regrette pas. Bien au contraire, je tiens à exprimer ici ma reconnaissance aux personnes qui m'ont fait découvrir la cryptographie et celles qui m'ont soutenu durant le doctorat, à la fois sur le plan professionnel et sur le plan personnel.

Tout d'abord, celui qui m'a fait confiance en m'accueillant dans son laboratoire, qui m'a donné les moyens pour réussir et qui m'a conduit jusqu'au doctorat, c'est Serge Vaudenay, à qui j'adresse ici mes sincères remerciements. Sa vivacité d'esprit, sa rigueur scientifique et son exigence ont été pour moi un exemple et m'ont permis de progresser durant les années passées à ses côtés.

Mes remerciements s'adressent également à ceux qui m'ont initié à la cryptographie, lorsque j'étais étudiant à l'université de Caen : Brigitte Vallée, sans qui je n'aurais jamais découvert cet univers passionnant, mais aussi Marc Girault, dont les enseignements ont incontestablement orienté mes choix. Je n'oublie pas non plus mes camarades d'études, en particulier Tony Gallon, Aline Gouget et Bruno Zanuttini. C'est avec Tony que j'ai poussé la porte du département d'informatique vers la fin de mes études de mathématiques ; sans le savoir, nous y entrions pour plusieurs années. Mes pensées s'adressent également à Faïza Abbaci, Fernando Aguiar, Stéphane Albin, Rémi Berthéas, Franck Gaultier, Camille Prime, Laurent Vercouter et Didier Vila, rencontrés il y a maintenant cinq ans et avec qui j'entretiens une profonde amitié.

C'est aussi en Suisse, où je me suis installé pour y préparer mon doctorat, que j'ai établi de nombreux contacts ; parmi eux, mes collègues et amis du Laboratoire de sécurité et de cryptographie (LASEC) de l'École polytechnique fédérale de Lausanne (EPFL). Je tiens à les remercier, pour avoir rigolé (souvent en se forçant un peu) lorsque je faisais mes blagues, mais aussi pour avoir entretenu une dynamique de réflexion et de travail propice à la production scientifique. Outre le patron, je remercie, par ordre d'apparition dans l'équipe : Pascal Junod, Philippe Oechslin, Brice Canvel, Lu Yi, Jean Monnerat, Thomas Baignères, Claude Barral, Julien Brouchier, Matthieu Finiasz, Martin Vuagnoux et Sylvain Pasini. Un merci tout particulier à Jean, qui a partagé mon bureau durant plusieurs années (ou peut-être est-ce moi qui ai partagé le sien) et avec qui j'ai pu échanger de nombreuses réflexions, tant sur la cryptographie que sur la généalogie de Pépin le Bref. Au sein de notre laboratoire, c'est aussi Martine Corval, notre secrétaire, que je tiens à saluer très chaleureusement, pour sa compétence, sa bonne humeur et sa sympathie. En cas de problème, on peut également compter sur l'épatante France Faille du Laboratoire de systèmes répartis, que je remercie pour sa disponibilité et sa grande gentillesse.

Bien que la préparation d'un doctorat ressemble parfois à une vie monastique, il est des moments où l'on prend plaisir à sortir de ce monde, à rencontrer des amis avec qui l'on peut parler de tout, sans ne jamais aborder les courbes elliptiques. C'est ainsi que mes moments de détente ont été principalement partagés durant cette thèse avec Thomas Alexandre, Cédric André, Christophe Bauer, Brice Canvel et Nathalie Guilbaud.

Tout au long de la thèse se forment également les connaissances professionnelles, par le biais des conférences. Comme dans un bar, y séjournent toujours les mêmes piliers, que l'on retrouve avec plaisir pour discuter de cryptographie ou, plus prosaïquement, pour boire un verre, jouer au billard ou visiter les Caraïbes. Il m'est impossible de nommer toutes ces personnes mais elles se reconnaîtront certainement. Certaines rencontres aboutissant à une publication commune, je remercie pour le travail accompli l'ensemble de mes co-auteurs : Étienne Dysli, Félix Gärtner, Rachid Guerraoui, Pascal Junod, Jean Monnerat, Philippe Oechslin, Thomas Peyrin, Serge Vaudenay et Marko Vukolić.

La rédaction de remerciements comporte un risque, celui d'oublier involontairement des personnes. Il y en a cependant que l'on ne peut oublier car, s'il n'y a pas de thèse sans candidat, s'il n'y a pas de thèse sans directeur, il n'y a pas non plus de thèse sans jury. Je tiens à remercier Moti Yung de Columbia University à New York, Jacques Stern de l'École normale supérieure de Paris, ainsi qu'André Schiper et Touradj Ebrahimi de l'EPFL, pour avoir accepté de participer à ce jury. Leur présence est pour moi un grand honneur.

Enfin, si les remerciements ressemblent parfois à un papillon nécrologique, la vie ne prend pas fin pour autant avec l'obtention du doctorat. Bien au contraire, le son du glas annonce le début d'une nouvelle aventure, et c'est ainsi que mes derniers remerciements s'adressent aux personnes qui m'ont proposé de poursuivre mes recherches au sein de leur groupe.

Acknowledgments

Cryptography is a really fascinating domain. A branch at the crossroads of several sciences, it brings to life mysterious anecdotes and stories. Through the ages, its techniques and objectives have evolved, but its heart, which makes it so exciting, did not change: cryptography is a game that consists of enigmas to be solved, plots to be disentangled, challenges to take up. The charm of cryptography is also the world that surrounds it, this world of cryptographers, *cryptopaths* one could say, made up of people each one more unique than the other. Pick any subject for conversation and the cryptopath will bring back it to his discipline. For example, install a new electronic coffee machine and the cryptopath will not be satisfied unless he succeeds in bugging it. Invite him to a restaurant where one can help ourselves at will and he will try a "plate replay attack" to feed the whole table. Offer him an electronic concert ticket and he will completely dissect it to analyze its security. It is undoubtedly an intense curiosity and a spirit more twisted than the average, which characterizes a cryptopath.

Taking the risk of not drinking anymore coffee, to no more be invited to the restaurant and to no more listen to Schubert, I chose to invest myself in this branch and do not regret it. Quite the contrary, I wish to express here, my gratitude to those people who made me discover cryptography and those who supported me during my PhD studies, both at the professional and personal levels.

First of all, I thank profoundly Serge Vaudenay, the person who trusted me by welcoming me in his laboratory and who gave me the resources to succeed. To me, his lively spirit and his scientific rigor served as an example and allowed me to progress during the years I spent with him.

I also wish to thank those who initiated me to cryptography, when I was student at the University of Caen: Brigitte Vallée, without whom I would never have discovered this enthralling universe, and also Marc Girault, whose teachings undoubtedly influenced my choices. I cannot forget my fellow students, in particular Tony Gallon, Aline Gouget, and Bruno Zanuttini. Towards the end of my mathematics studies, it is with Tony that I timidly opened the computer department's door, without knowing that we would stay there. My thoughts are also with Faïza Abbaci, Fernando Aguiar, Stephan Albin, Rémi Berthéas, Franck Gaultier, Camille Prime, Laurent Vercouter, and Didier Vila, whom I met five years ago and with whom I share a deep friendship.

It is also in Switzerland, where I settled to work on my PhD, where I established many contacts; among them, my colleagues and friends from the Security and Cryptography Laboratory (LASEC) of the Swiss Federal Institute of Technology (EPFL). I wish to thank them, to have laughed (often while forcing themselves a bit) when I cracked my jokes, and also to have maintained a working and thoughtful environment favorable to scientific production. In addition to the boss, I thank, in order of appearance in the team: Pascal Junod, Philippe Oechslin, Brice Canvel, Lu Yi, Jean Monnerat, Thomas Baignères, Claude Barral, Julien Brouchier, Matthieu Finiasz, Martin Vuagnoux, and Sylvain Pasini. A special thanks to Jean, with whom I shared my office during several years (or perhaps is it me who shared his office) and with whom I could exchange many thoughts, as much on cryptography as on the genealogy of the French King "Pépin le Bref". Inside our laboratory, I also wish to warmly thank Martine Corval, our secretary, for her competence, good mood and sympathy. In case of problems, I could also count on the wonderful France Faille from the Distributed Systems Laboratory, and I thank her for her availability and warm kindness.

Although life during PhD studies sometimes resembles monk's life, it is filled with moments where one takes pleasure to leave this academic world, to meet friends with whom one can speak about everything, without ever approaching the elliptic curves. It is in this way that my relaxation moments were lived throughout my thesis work and mainly shared with Thomas Alexandre, Cédric André, Christophe Bauer, Brice Canvel, and Nathalie Guilbaud.

Throughout the thesis, one also creates professional contacts, by means of the conferences. Just like in a bar, we always find the same people, and it is a pleasure to discuss cryptography with them, or more prosaically, share a drink, play pool, or visit the Caribbean Islands. It is impossible for me to name all these people but they will certainly identify themselves. Certain meetings lead to a common publication; I thank all my co-authors for the accomplished work: Étienne Dysli, Félix Gärtner, Rachid Guerraoui, Pascal Junod, Jean Monnerat, Philippe Oechslin, Thomas Peyrin, Serge Vaudenay, and Marko Vukolić.

Writing acknowledgments involves a risk, that is, involuntarily forgetting people. However, there are some we cannot forget because, if there is no thesis without a candidate, if there is no thesis without a supervisor, there is also no thesis without a committee. I would like to thank Moti Yung from Columbia University in New York, Jacques Stern from École normale supérieure in Paris, as well as André Schiper and Touradj Ebrahimi from EPFL, for agreeing to participate in this thesis committee. For me, their presence is a great honor.

Lastly, if the acknowledgments sometimes resemble a death notice, life will not end after obtaining the PhD. Quite the contrary, the death knell's sound announces the beginning of a new adventure, and it is in this manner that my last thanks are addressed to the people who offered me to continue my research within their group.

Foreword

This PhD thesis presents my research results on fair exchange and Radio Frequency Identification (RFID). In the first part, which includes Chapters 1 to 4, my studies related to fair exchange [20, 22, 28, 29, 30] are presented. The second part includes Chapters 5 to 9, which exposes my work on RFID protocols [15, 16, 19, 26, 27], as well as Chapter 10 that introduces new results on the time-memory trade-offs [23, 24], which find applications to RFID.

Fair exchange stems from a daily life problem: how can two people exchange objects (material or immaterial) fairly, that is, without anyone being hurt in the exchange? More formally, if Alice and Bob each have objects m_A and m_B respectively, then the exchange is fair if, at the end of the protocol, both Alice and Bob have received m_B and m_A respectively, or neither Alice nor Bob have received the expected information, even partially. The difficulty fundamentally arises from the exchange's non-atomicity: the first person, who receives the other's object, can stop the exchange and keep both objects. A typical example is the mail-order selling. In such a case, the salesman does not want to send the item first out of fear of not being paid and, reciprocally, the customer does not want to send the payment first out of fear of not receiving the expected item. The difficulty is naturally amplified in the case of exchanges on computer networks: on the one hand for technical reasons (the protocol can be unexpectedly stopped), on the other hand because of the difficulty in identifying the correspondent, or even prosecuting him in case of conflict. The fact that in this case, the object is immaterial – one will then speak of "information" – and can thus be duplicated at will, does not modify the problem, but opens, as we will see further, interesting perspectives to solve it. Although the fair exchange problem can be easily stated, to find a solution is a challenge without hope. Indeed, Even and Yacobi [71] showed in 1980, that the fair exchange between only two parties is impossible without adding assumptions on the problem.

In **Chapter 1**, we provide an introduction to the fair exchange: we point out the various approaches that were explored in this field, we describe the differences between the two-party fair exchange and the multi-party fair exchange, we formalize the participants' properties, the communication channels and the items to be exchanged, and finally, we provide examples of fair exchange protocols.

In **Chapter 2**, we propose a probabilistic solution, that consists in attaching to each participant, a *guardian angel*. A guardian angel is a security module, conceived by a trustworthy authority, whose behavior cannot deviate from the established rules. We introduce a synchronization protocol called *Kit-in-Touch* (KiT), which constitutes the basic building block of our fair exchange protocol. We show that by using the KiT protocol, the guardian angels can ensure the fairness of the exchange with a probability as close to 1 as desired, implying however a communication complexity cost. We prove that no protocol can solve the synchronization problem with a probability better than inversely linear in the number of rounds. In that sense, our protocol is optimal.

Then in **Chapter 3**, we use results from the distributed algorithms' consensus problem to extend the guardian angels model to the multi-party case. We provide a synchronous distributed protocol, called *Gracefully Degrading Fair Exchange* protocol, aimed at exchanging digital items among untrusted parties, each hosting a guardian angel. As an underlying building block, our algorithm uses an early stopping subprotocol that solves in the general omission failure model, a specific variant of consensus called *biased consensus*. The algorithm provides the following two complementary features: (1) If a majority of the parties are honest, then the algorithm deterministically guarantees the completeness, fairness, and termination properties. This is optimal in terms of resilience: we indeed show that, even in a synchronous model with guardian angels, no deterministic algorithm solves fair exchange if half of the parties are dishonest. (2) If at least half of the parties turn out to be dishonest, then our algorithm degrades gracefully in the following sense: it still guarantees the completeness and termination properties, as well as ensures that the probability of violating fairness can be made arbitrarily low.

Lastly, in **Chapter 4**, we propose a second approach that consists in no longer considering the exchange in an isolated manner, but to replace it in its context, in the heart of a network, where each person in the pair will have a few honest neighbors. In this framework, fairness can lie on these neighbors, who will be solicited only in the case of a conflict during the exchange. For that, our two-party protocol relies on a publicly verifiable secret sharing scheme. Our protocol can be seen as a kind of distributed optimistic protocol. Beyond the fact that this approach is novel, it allows ensuring privacy: except the two main protagonists, no other participant gets any information about the expected items. This is a great improvement compared to previous optimistic fair exchange protocols where we usually assumed that the expected items are revealed to the trusted third party in case of conflict.

Our results on fair exchange led to four papers and one technical report. The guardian angel model has been introduced in the MC2R journal [28] and published at WISA'03 [29]. It is a joined work with Serge Vaudenay. The generalization to the multi-party case has been presented during an Ecrypt Workshop [21] and published at EDCC'05 [22]. An extended version is available in a technical report [20]. It is a joined work with Rachid Guerraoui, Félix Gärtner, and Marko Vukolić. Results on the fairness in a pretty honest neighborhood [30], a joined work with Serge Vaudenay, have been published at ACISP'05. Finally, a brief introduction to fair exchange has been published in the French magazine *MISC* [14].

In the second part of this thesis, we deal with Radio Frequency Identification (RFID). This technology allows identifying objects or subjects with neither physical nor visual contact. For that, a small transponder is placed on the object. Even though it has only been a few months since it is covered by the media, this technology is fundamentally not new. However, the boom that RFID enjoys today rests on the ability to develop very small and cheap transponders called "electronic tags". We focus on these low cost tags, which only offer weak computation capacities. Indeed, the most advanced ones are only able to use symmetric cryptography. Even if public key cryptography became accessible in the future on this type of tag, one should not halt research based on this assumption. This is because tags using symmetric cryptography will always have their place, since the decision makers prefer reducing costs rather than increasing the tags' capacities.

In **Chapter 5**, we provide a thorough introduction to the RFID technology. We present daily life examples, we describe the RFID system's architecture and the tag's characteristics. Then, we discuss the differences between the notions of identification and authentication. This last point is never addressed in the literature while in our opinion, it is fundamental.

In **Chapter 6**, we introduce problems related to security and privacy that threaten RFID. We also present measures that are deployed to reduce these threats. In particular, we describe the two large families of RFID protocols that allow preserving privacy. The first consists of protocols where the reader actively helps the tags to protect their privacy. The second family consists of protocols where the tags themselves deal with this problem. We also introduced the concept of (malicious) traceability and we point out and formalize the link between traceability and communication model. In particular, we show that collision-avoidance protocols usually endanger privacy. We illustrate our view by pointing out weaknesses in protocols that are already implemented in today's tags.

Chapter 7 deals with protocols where the reader actively helps the tags to protect their privacy. Six protocols are analyzed and, for each of them, we propose attacks that violate the tag bearers' privacy.

Chapter 8 deals with protocols where the tags refresh their identifiers by themselves, without the reader's help. Six protocols are presented. This chapter also stresses on the complexity issues on the system's side that arise during the identification process. We show that the identification of a single tag requires that the system carry out an exhaustive search in its database, which is impossible to practically implement in large scale applications.

In **Chapter 9**, we describe Molnar and Wagner's technique that makes it possible to reduce the identification complexity from $O(n)$ to $O(\log n)$ where n is the number of tag in the system. We show that this technique degrades privacy of the system's tags if the adversary is able to tamper with one or more tags. Thus, this technique is only a compromise between privacy and complexity. We then propose another technique based on time-memory trade-off, which practically allows achieving the same performances as Molnar and Wagner's technique, but without degrading privacy.

Lastly, our research aimed at improving tag identification led us to more general research on time-memory trade-offs. Thus, in **Chapter 10** we show an approach that allows reducing the time wasted due to false alarms. This technique is based on *checkpoints*, that is, positions on the chains where tests are carried out. These tests, typically parity checks, allow detecting probabilistically if two chains leading to the same end have collided. We supply a thorough analysis of rainbow tables, proposed by Oechslin in 2003, which had never been done yet. We apply our checkpoint technique to this variant of the trade-off and show that it is better, up to a given threshold, to use memory for storing checkpoints instead of using memory for storing chains. We also supply a few implementation tips that greatly improve the performance of the trade-off in practice.

Our results on radio frequency identification and time-memory trade-off led to five papers and two technical reports. Results on the multilayered approach of the traceability have been published at Financial Cryptography 2005 [26]. The technique of reducing the identification complexity based on time-memory trade-off has been published at PerSec'05 [27]. Both are joined works with Philippe Oechslin. The attack against Juels and Pappu's banknote protection scheme has been published at Cardis 2004 [15], where it won the best student paper award. The attack [19] against Molnar and Wagner's protocol is found in the proceedings of SAC'05. It is a joined work with Étienne Dysli and Philippe Oechslin. An overview of this work has been presented during an invited talk in an Ecrypt Workshop [18]. The work with Pascal Junod and Philippe Oechslin on the checkpoints in time-memory trade-off has been presented at Indocrypt 2005 [23]. An extended version is available in a technical report [24]. Finally we propose in a technical report [16], a formalization of the adversary model. This work has not yet been published in a conference and is therefore not presented in my thesis. My activities in RFID also include the management of a web-based lounge [13] devoted to security and privacy in RFID systems.

Some of my papers of lesser importance and of more exotic nature are not presented in this thesis. Among them, the paper [25] has been published at Indocrypt 2004. This joined work with Jean Monnerat and Thomas Peyrin improves some results proposed by Muir and Stinson on the non-adjacent form representation of integers. Also, [17] has been published at Financial Cryptography 2005. This work presents an attack against the lightweight micropayment scheme suggested by Jakobsson, Hubaux, and Buttyán. Finally, my research and educational activities led me to publish, in collaboration with Pascal Junod and Philippe Oechslin, a book in network security entitled "Sécurité informatique, Exercices corrigés" (Vuibert, 2004).

Part I

Fair Exchange

Introduction to Fair Exchange

CHAPTER ONE

Currently, electronic exchange of information has become a common matter. For example, in e-commerce, it is important that the exchanges be made in a fair way, that is to say, none of the parties involved in the exchange feel injured. In other words, a) a communicating entity, having sent an item to another entity, receives in response the corresponding expected item, or b) none of the entities involved in the exchange receive anything valuable.

Many variants of fair exchange exist that take advantage of the nature of the items being exchanged. For example: the exchange of digital signatures on a document (digital contract signing protocols), the exchange of information against an acknowledgment (certified email protocols), the exchange of information associated with the evidence of origin against evidence of receipt (non-repudiation protocols), and the exchange of digital goods of similar value. The last example is the most general case of fair exchange, and is the focus of this thesis. Beyond the nature of the item itself, protocols can be classified according to the required properties on the communication channels (see Section 1.3.1), on the participants (see Section 1.3.2), or on the properties of the items being exchanged (see Section 1.3.3). As a result of all these variations, there exists a large literature linked to fair exchange. Some excellent reference documents are Zhou's thesis [186] and Markowitch's thesis [127] both devoted to non-repudiation protocols; Asokan's thesis [5], which can be seen as the revival of the fair exchange problem with the development of the notion of optimistic protocols; Schunter's thesis [159] which focuses on this latter type of protocol; and finally Kremer's thesis [123], which deals with a formal analysis of fair exchange protocols and gives a good historical approach to the definition of fair exchange.

1.1 Fair Exchange Primer

Originally, the basic exchange scheme in which participants send their items one after the other is obviously unfair. Indeed, neither the originator nor the recipient wants to be the first to send his item because the first receiver can disrupt the protocol without sending his own item. Even and Yacobi [71] proved in 1980 that fair exchange between only two entities is impossible.

First (partial) solutions to this problem appeared in the eighties with the *gradual* fair exchange protocols (e.g., [34, 44, 49, 56, 59, 69, 70, 109, 140, 154, 164]). The technique is based on a gradual release of knowledge: each entity alternately transmits successive bits of the item to be exchanged until the last bit of each item is sent. If the protocol aborts, each participant can retrieve the missing bits, thus inducing a computation cost. Hence, the fairness of gradual protocols relies on the originator and the recipient having approximately the same computational power and the lengths of the items being equal. However, what few authors point out is that it is not always possible for the injured party to recover the desired information, independently of its computational power. In fact, being able to recover the missing item greatly depends on the type of item. For example, if the item is a PIN giving access to a smart card, the injured party will perhaps have only a limited number of trials in order to recover the missing item. If the number of unsuccessful attempts is attained, the fairness is lost. Another difficulty is, the sender must persuade the other party that the bit he sends really enables the recovery of the expected item. In order to reduce the loss of fairness during the execution of the protocol, in 1983 Tedrick [165, 166] suggested that instead of exchanging bits of information, the base unit could be a fraction of a bit. However, the smaller the unit of information, the greater is the protocol's communication complexity.

Protocols that no longer rely on the participants having the same computational capacity have been put forward. Here, the fairness is *probabilistic* and not gradual. Markowitch and Roggeman [129] in 1999, then Mitsianis [133] in 2001, suggested a non-repudiation protocol where the recipient receives several messages but does not know at first sight which transmission contains the information to be exchanged. The probability of guessing that the transmission includes the right message can be made arbitrarily small. In order to decrease this probability, the number of messages has to be increased. The main disadvantage of this method is that it assumes that the originator of the exchange knows the computational capacity of the recipient. Indeed the fairness relies on the fact that the recipient is not able to decrypt on-the-fly the messages it receives during the execution of the protocol. Boneh and Naor [46] also explored this possibility in 2001 by putting forward a protocol which makes it difficult for the recipient to parallelize the computations.

Other approaches involve an online *trusted third party* (TTP) that acts as a mandatory intermediary between the participating entities (e.g., [31, 57, 58, 60, 81, 146, 149, 183, 186, 189, 190]). Here, the fairness is deterministically ensured, but the major drawback of this approach is the bottleneck created at the TTP, as well as from a communication, computation and storage point of view.

Instead of having an online TTP, it is possible to have an *offline* TTP which does not intervene in the exchange of the items. An important step was achieved in 1996 when Asokan [5, 7, 9, 10, 11] developed an approach where the TTP is only necessary in case of a conflict between the participants or, more generally, when a problem arises during the execution of the protocol. This so-called *optimistic* approach, initially suggested by Bürk and Pfitzmann [50] in 1990, assumes that the participants are usually honest. Almost all two-party fair exchange protocols that are proposed nowadays rely on this model (e.g., [32, 85, 130, 150, 159, 172, 187]). However, this model requires fairly strong assumptions, not only on the items being exchanged, as we will see in Section 1.3.3, but also on the communication channels. Indeed, all these protocols suppose that both parties have access to a TTP though a reliable channel, implying that any request to the TTP will eventually be addressed. This is quite a strong assumption, for example in mobile networks, because an adversary may control all communications in one cell. As pointed out by Ray and Ray in [148], it is important to find alternatives to existing protocols, alternatives that do not rely on the presence of a centralized TTP. We will propose such alternatives in the next chapters.

1.2 From Two-Party to Multi-Party

1.2.1 Two-Party Case

Several (different) definitions for fair exchange exist in the literature. Most of them are context-dependent. Some efforts have been made in order to converge towards a unified definition. More details about the different existing definitions can be found in [123] where Kremer provides a historical approach to the definition of fairness, and in [90] where Gärtner, Pagnia, and Vogt explore the definition of fairness in the field of concurrency theory, compared to the common one used in electronic commerce. Below we provide a rather universal definition based on well-known existing definitions (e.g., [10, 90, 123, 188]), which rely on three properties: *completeness*, *fairness*, and *termination*.

Definition 1 (two-party exchange protocol). *An exchange protocol between two parties A and B is a protocol in which A and B possess some items m_A and m_B respectively and aim at exchanging them. We say that the protocol ensures:*

- completeness *if A gets m_B and B gets m_A at the end of the protocol when there is no malicious misbehavior;*
- fairness *if the protocol terminates so that either A gets m_B and B gets m_A, or A gets no information about m_B and B gets no information about m_A;*
- termination *if A and B eventually terminate the protocol.*

Definition 1 corresponds to what purists call “fairness” and which is sometimes referred to as *strong* fairness. However, most of fair exchange protocols rely on gathering evidence that during the protocol execution, the other party actually participated. This evidence can later be used for dispute resolution in a court of law. The dispute resolution phase is not a part of the protocol. Such protocols are said to ensure *weak* fairness.

1.2.2 Multi-Party Case

In the literature, the majority of the proposed fair exchange protocols are two-party protocols. Since less than a decade, efforts have been made to generalize some of these two-party protocols to the case of n participants. Considering an exchange protocol with more than two entities implies a lot of new and interesting issues: we will see that different topologies have to be considered, the manner of conceiving fairness may vary, and the need for confidentiality may appear. Below, we generalize the two-party fair exchange problem to the multi-party case. Firstly, we bring forth the definition of multi-party fair exchange without considering any particular exchange topology.

Definition 2 (multi-party exchange protocol). *An exchange protocol between several parties is a protocol in which each party possesses some item and aims at exchanging it. We say that the protocol ensures:*

- completeness *if every party gets its expected items at the end of the protocol when there is no malicious misbehavior;*
- fairness *if the protocol terminates so that either every party gets its expected items, or no party gets any information about the other items;*
- termination *if every party eventually terminates the protocol.*

The fairness definition used here requires that at the end of the protocol, *all* participating entities have got their item or none of them gained any valuable information. This implies that a single entity has the power to stop the protocol for all other entities.

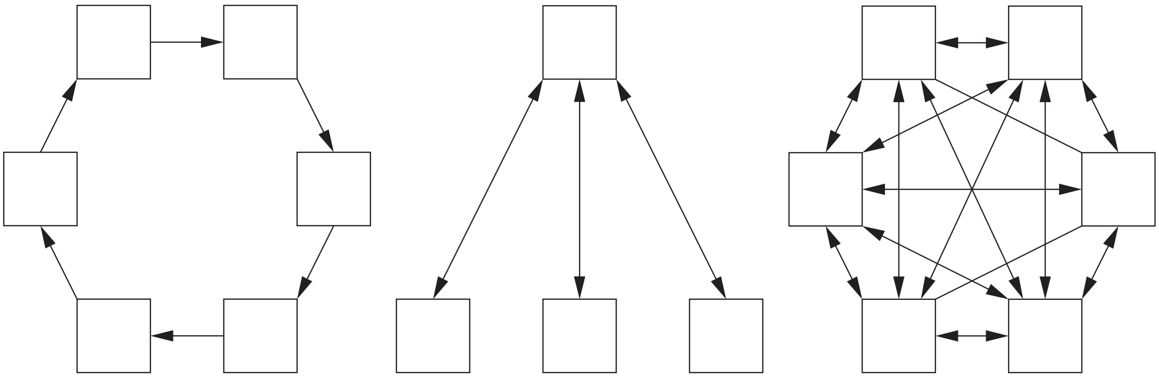
Switching to a multi-party framework also implies certain new specific security risks that are not of concern in a two-party case. For example, we may encounter the problem of exclusion [95] of communication entities. A participant is said to be *excluded* from an exchange if it has taken part in the setup phase (during which the entities willing to participate in a given exchange execution agree on, which other entities will take part in this exchange, on the items to be exchanged and on how these items will be exchanged during the exchange phase) but a group of other participants involved in the exchange has prevented it from participating in the exchange phase. A multi-party exchange protocol is said to be *weak exclusion-free* if any excluded participant from an execution of this protocol is able to prove to an external adjudicator that it has been excluded. In the same way, a multi-party exchange protocol is said to be *strong exclusion-free* if, at the end of an execution of the protocol, there are no excluded participants. Therefore, in an exchange protocol not providing strong exclusion-freeness any honest participant has to trust the remaining participants for not being excluded from the exchange.

Another specificity of multi-party fair exchange protocols (e.g., digital contract signing) is the possible *abuse-freeness* property that ensures the impossibility for a single entity, at any point in the protocol, to be able to prove to an outside party that it has the power to terminate or successfully complete the protocol [86]. To illustrate this property, let us consider an entity Alice who wants to sign a digital contract with the highest offerer. Suppose Bob was chosen and begins a non abuse-free digital contract signing protocol with Alice. One way for Alice to increase the offer consists in proving to another offerer, Oscar, that she can successfully complete the protocol with Bob. In other words, Alice proves to Oscar that Bob is committed to the contract. Therefore, Oscar will be convinced of making a higher offer (Alice could then stop the protocol with Bob and restart the protocol with Oscar). With an abuse-free contract signing protocol such a behavior is no longer possible.

1.2.3 Topologies

When considering the migration from a two-party fair exchange protocol to $n > 2$ communicating entities, we need to define the exchange topology. Therefore, we describe the commonly used topologies.

We could assume a protocol where the exchange is realized following a *ring topology* (see Figure 1.1a): each entity offers its information to the next entity in the (oriented) ring and requires an information from the previous one [33, 82].



(a) Ring topology (b) One-to-many topology (c) Many-to-many topology

Figure 1.1: Common topology graphs of exchanges

In multi-party certified email protocols and multi-party non-repudiation protocols it does not make sense that one entity providing an information, receives (possibly indirectly) a receipt from someone else than the corresponding recipient. All recipients expect the same information, while the sender waits for an acknowledgment from each recipient. A recipient does not want any evidence from the other recipient. Therefore, the most natural generalization seems to use a *one-to-many topology* (see Figure 1.1b), where one entity sends a message to $n-1$ receiving entities who respond to the sender [128].

Another possible instance can be studied by considering multi-party digital contract signing protocols with n participants. In that case, each participant aims at obtaining the $n-1$ signatures of all other participants. This is an example of a possible use of a *many-to-many topology* (see Figure 1.1c).

Finally, the exchange may rely on a more general topology where each entity may require items from a set of entities and offer items to another set of entities [6, 8]. In that case, the topology graph is a (connected and oriented) sub-graph of the many-to-many topology graph.

1.3 Description of the Environment

1.3.1 Communication Channels

The properties of a channel are generally not intrinsic to the physical channel, but they are related to additional mechanisms, for instance tunneling techniques. Such techniques guarantee packet security, which relates to the security of individual messages as well as session security, which relates to the security of the whole communication session. Packet security typically consists of:

- *Confidentiality*: the sender is assured that *only* the correct receiver will get the packet.
- *Authentication:* the receiver is assured that something has been sent by the correct sender.
- *Integrity*: the receiver is assured that the packet it receives is exactly the one which was sent. When integrity and authentication are combined, as it usually is (and implicitly) the case, the receiver is assured that *this* packet was sent by the correct sender.
- *Liveliness*: the sender is assured that the packet will eventually be delivered to the correct receiver.
- *Timeliness*: the sender is assured that the packet will be delivered to the correct receiver within a time-bounded delay.

Session security consists of:

- *Sequentiality*: for any party, at any time when receiving a packet, the observed transcript of the protocol is equal to the transcript observed by the other party, at least at some other time in the past. Thus, no packet can be replayed, the packets order cannot be modified, and packets cannot be dropped, unless if all subsequent packets are dropped as well (or equivalently if the two entities are disconnected).
- *Safe termination*: the sender and the receiver are assured that they both close the session and are mutually aware regarding the completion of the protocol.

1.3.2 Participants

Two-party fair exchange protocols basically involve two parties A and B that want to exchange items m_A and m_B. These parties are called the *main* participants. Conventionally, A represents the *originator* of the exchange while B represents the *recipient.* A participant who follows the protocol rules is said to be *honest*; otherwise, it is said to be *dishonest.* It has been shown in [71] that protocols involving only main participants cannot ensure fairness with probability 1. Consequently, other parties are usually involved in the exchange in order to ensure or restore fairness. These parties are called *external* participants and usually consist of a TTP either *inline*, *online*, or *offline* (*optimistic*) defined as follows:

- *Inline*: the TTP is involved in the transmission of every message.
- *Online*: the TTP is involved in every protocol execution.
- *Offline*: the TTP is sometimes involved, but not in every protocol execution. It is said to be *optimistic* if it is only involved under exceptional circumstances due to the misbehavior of a main participant or when a network problem occurs.

The case where both main participants collude together is a non-sense in fair exchange protocols. More generally, the case where both main participants are dishonest is not relevant since fairness cannot be ensured in such a frame. On the other hand, the case where one participant colludes with the third party is also not considered because almost all existing two-party fair exchange protocols assume that the third party is trusted. An exception can be found in [81] where the third party is only *semi-trusted*: it may misbehave for its own benefit but it does not conspire with one of the main participants. Therefore, common attacks in a two-party fair exchange consist of one main participant who is honest and another who tries to misbehave by deviating from the protocol. The latter's goal is to obtain the expected item without disclosing its own item.

The analysis is much more tricky in the multi-party case since dishonest participants can conspire. The reason for a participant's misbehavior is also more tricky and is related to the fairness definition as described in Section 1.2. Consider for instance three participants A, B and C. A agrees to perform a fair exchange with B but disagrees to perform a fair exchange with C; unfortunately, B agrees to perform a fair exchange with A if and only if C is also involved in the exchange. So A will accept the deal but will try to misbehave in order to exclude C.

1.3.3 Items Properties

In general, the exchanged items are bit strings that can be transferred to or stored by other parties. At the end of the interaction between the parties, each of them must be able to verify that the received item is the expected one. For this, we assume that the parties agree in advance on a mathematical description of the items to be exchanged. At the end of the exchange, each party can then verify that the received item matches the expected one. If this is not the case, the conflict can be resolved by calling on an external participant, if one exists. It is clear that if the mathematical descriptions of the items do not match their authentic human-readable descriptions, the conflict can only be resolved in front of judicial authorities. However, this type of conflict is out of the scope of our work.

In addition to the nature of the items (document, signature, proof of non-repudiation, etc.), items can be classified according to other properties, which must be considered when the fairness relies on a TTP. Firstly, the items to exchange can be assumed to be *idempotent* or not, meaning that receiving these items several times has the same effect as receiving it once. Secondly, Asokan, Schunter, and Waidner [9] have shown that the items should respect the properties of either *generatability* or *revocability*, defined as follows:

- *Generatability*: the TTP has the ability to regenerate the expected item on its own.
- *Revocability*: the TTP has the ability to invalidate the expected item.

Generatability of an item can be used to resolve conflicts during an exchange: if a participant does not receive the expected item, he can ask the TTP to generate this missing item. This type of generatability meets the notion of strong fairness and is hence called *strong* generatability. If the item generated by the TTP is identical to the item transmitted by its owner, the TTP is said to be *transparent* or *invisible* as the item itself does not reveal whether the TTP was involved or not. Such an approach, initially suggested by Micali in 1997, is particularly relevant in trading applications where intervention of the TTP, possibly due to failures, gives rise to bad publicity. When the TTP can fail while generating the missing item but can always determine which main participant cheated, we speak of *weak* generatability.

Revocability of an item can be used to invalidate an item that has already been delivered so that it becomes useless to its receiver. It is well-suited to fair purchases, where electronic money can be revoked, if the expected item is not delivered. Such a revocability is said to be *strong*. A variant is the notion of *weak* revocability addressed by Vogt in [172]: the TTP can try to revoke the item, but it may fail. However, this proves that the item has really been delivered to the party that expected it. As pointed out by Pagnia, Vogt, and Gärtner in [141], this property matches the notion of non-repudiation of receipt [186]. Revocable items have been much less studied than generatable items. However, it is often easier to revoke an item than to regenerate it, for example in electronic payment systems.

An additional property, which is achieved by certain protocols, is the *time-sensitive* property, meaning that the value of the item (from the receiver's point of view) can develop over time, e.g., a concert ticket. For example, certain items become useless after an expiry date. Vogt, Pagnia, and Gärtner [173, 174] put forward a time-sensitive fair exchange protocol between a customer and a merchant, where the customer can reject the expected item (without violating fairness) if it arrives too late. More details on this protocol are given in Chapter 2.

1.4 Example of Two-Party Fair Exchange

Below, we give an example of two-party fair exchange, namely the optimistic two-party fair exchange protocol suggested by Asokan *et al.* [7, 9, 10], which today is the reference model in terms of optimistic protocols. It consists of three sub-protocols: the exchange protocol where the main participants exchange their items; the abort protocol which can be executed by the originator in order to give up the exchange; and the recovery protocol which can by launched by either the originator or the recipient when the fairness has been lost. When there is neither misbehavior nor failure, only the exchange protocol is executed. The TTP, denoted by T below, is only involved in the abort and the recovery protocols.

During the first phase of the protocol (steps 1 and 2 below), the main participants A and B exchange commitments on the expected items. The items themselves are exchanged during the second phase (steps 3 to 5 below).

The commitment scheme relies on two functions, Commit and VerifyCommit. Given a string m and a key k, Commit generates a commitment $c = \mathsf{Commit}(m, k)$. The requirements on a commitment are that nobody can modify its content without invalidating it and that nobody can obtain any information about its content unless the committer explicitly opens it [142]. Intuitively, the idea is that, after having committed on the message to exchange, the committer cannot change it without being detected by the receiver. More precisely, given a string m, a key k, and a commitment c, VerifyCommit outputs "true" if and only if $\mathsf{Commit}(m, k) = c$.

We assume that each participant P (A, B, and T) has a signing key S_P and a verifying key V_P. They agree on a signature scheme (with message recovery) whose signature and verification functions are denoted Sign and VerifySign respectively. Finally, we consider a public collision-resistant hash function h.

Let i_A (resp. i_B) be the item own by A (resp. B), and d_A (resp. d_B) a mathematical description of i_A (resp. i_B). At the beginning of the exchange, A and B agree on the descriptions. The protocol exchange is described below and depicted in Figure 1.2.

- *Step 1:* A picks a random key k_A and computes $c_A = \mathsf{Commit}(i_A, k_A)$. Then it picks a random number r_A, computes $h_A = h(r_A)$ and finally sends B the message

$$me_1 = \mathsf{Sign}_{S_A}(V_A, V_B, T, c_A, h_A, d_B, d_A).$$

- *Step 2:* B checks that the received signature is valid, using VerifySign and VerifyCommit, and that the received descriptions match the expected items. If B decides to continue the exchange, it generates a random key k_B and picks a random number r_B. Then it computes $c_B = \mathsf{Commit}(i_B, k_B)$ and $h_B = h(r_B)$. Finally it sends A the message $me_2 = \mathsf{Sign}_{S_B}(me_1, c_B, h_B)$.

- *Step 3:* A checks the validity of the received message, using VerifySign and VerifyCommit. If it decides to give up, it runs the abort protocol. Otherwise, it sends $me_3 = i_A, k_A$ to B.

- *Step 4:* B executes the recovery protocol or sends $me_4 = i_B, k_B, r_B$ to continue the protocol.

- *Step 5:* A executes the recovery protocol or sends $me_5 = r_A$ to continue the protocol.

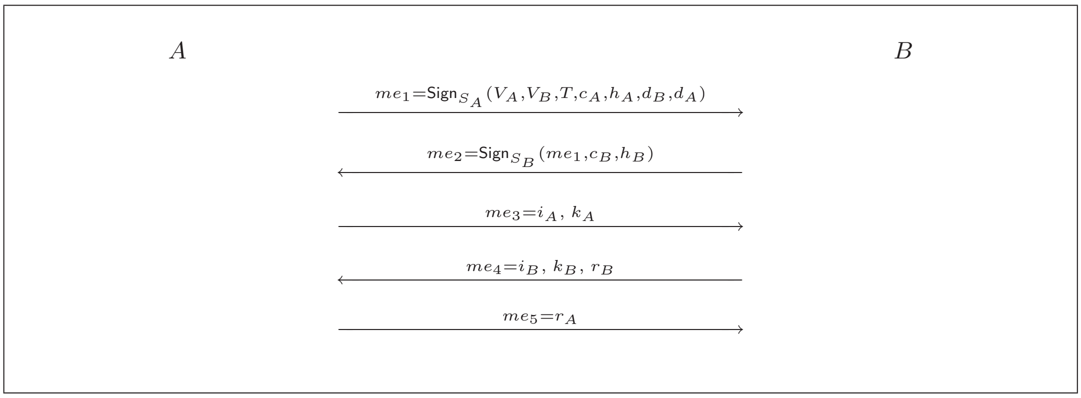

A
B
$me_1{=}\mathsf{Sign}_{S_A}(V_A,V_B,T,c_A,h_A,d_B,d_A)$
$me_2{=}\mathsf{Sign}_{S_B}(me_1,c_B,h_B)$
$me_3{=}i_A,\ k_A$
$me_4{=}i_B,\ k_B,\ r_B$
$me_5{=}r_A$

Figure 1.2: Asokan, Shoup, and Waidner's fair exchange: exchange protocol

The protocol abort is depicted in Figure 1.3. In the first message, A sends a request to abort. If B has already run recovery, then T informs A to execute recovery instead of abort. Otherwise, T marks the exchange as aborted and issues a receipt.

The recovery protocol is depicted in Figure 1.4, A being the originator. This protocol can also be called by B; in this case, the roles of A and B would be inverted. If the participant who is not the originator of recovery does not collaborate, the trusted third party signs a receipt to the originator. This receipt is a proof that fairness has been violated by the other participant. This proof can be used in a court of law.

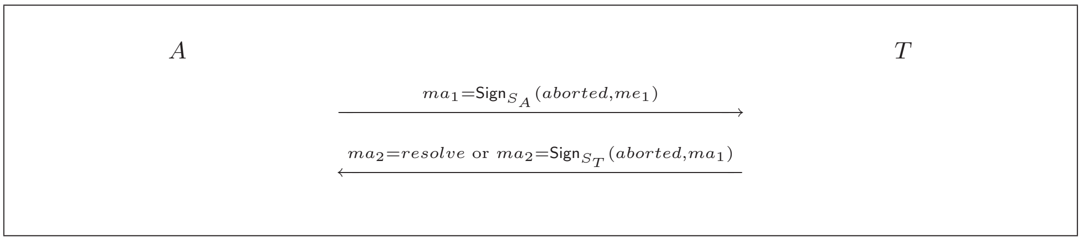

A
T
$ma_1{=}\mathsf{Sign}_{S_A}(aborted,me_1)$
$ma_2{=}resolve$ or $ma_2{=}\mathsf{Sign}_{S_T}(aborted,ma_1)$

Figure 1.3: Asokan, Shoup, and Waidner's fair exchange: abort protocol

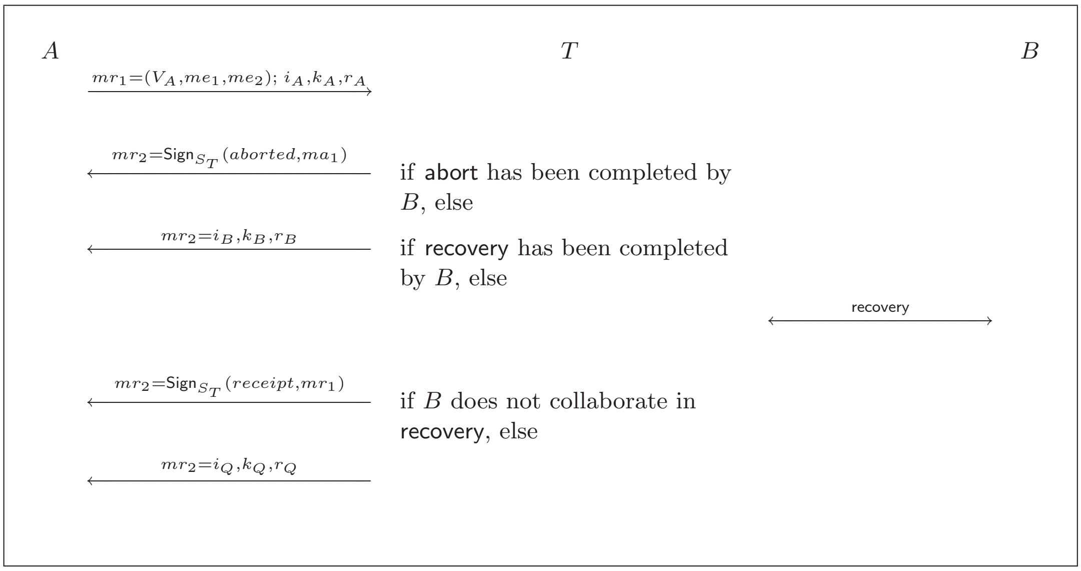

A
T
B
$mr_1{=}(V_A,me_1,me_2);\ i_A,k_A,r_A$
$mr_2{=}\mathsf{Sign}_{S_T}(aborted,ma_1)$
if abort has been completed by B, else
$mr_2{=}i_B,k_B,r_B$
if recovery has been completed by B, else
recovery
$mr_2{=}\mathsf{Sign}_{S_T}(receipt,mr_1)$
if B does not collaborate in recovery, else
$mr_2{=}i_Q,k_Q,r_Q$

Figure 1.4: Asokan, Shoup, and Waidner's fair exchange: recovery protocol

1.5 Example of Multi-Party Fair Exchange

We present in this section the (one-to-many) multi-party fair exchange protocol suggested by Zhang, Shi, and Merabti in [184]. In this protocol, the fairness does not rely on a trusted third party but on the honesty of the main participants themselves. More precisely, it consists in sharing the items to exchange between all the entities. In that sense, it differs from all the other multi-party fair exchange protocols.

When the protocol is launched, each participant P_i ($1 \leq i \leq n$) possesses an item m_i that it should send to the other $n-1$ participants. Let n' denote the number of potentially dishonest participants. We have:

$$0 \leq n' < \left\lceil \frac{n}{2} \right\rceil .$$

The exchange protocol (steps 1 to 4 below) consists of encrypting each item m_i with a session key s_i and then sharing these keys among all the participants by using a $(n'+1,n)$-secret sharing protocol [94]. Dishonest participants are thereby not able to reconstruct keys without the assistance of at least one honest party; on their side, honest parties can reconstruct the expected keys without the contribution of dishonest participants. In case of conflict, i.e., some participants does not receive enough shares to recover their expected secrets, the injured participant executes the recovery protocol (step 5 below).

- *Step 1 (item distribution):* each participant P_i chooses a session key s_i in order to encrypt the item m_i. P_i then computes the shares s_{ij} $(1 \leq j \leq n,\ i \neq j)$ of s_i and distributes them along with m_i encrypted with s_i to all other participants in such a way that only P_j is able to read s_{ij}.

- *Step 2 (acknowledgment broadcasting):* each participant acknowledges the receipt of the expected shares and items.

- *Step 3 (willingness announcement):* each participant P_i having received all the acknowledgments informs the other parties that it is able to reveal the shares s_{ji} $(1 \leq j \leq n,\ j \neq i)$.

- *Step 4 (secret revealing):* each participant P_i having announced during the third step his ability to reveal the shares and having received a minimal number of such announcements (at least $2n'$), sends these shares s_{ji} $(1 \leq j \leq n,\ j \neq i)$ to all the participants of the exchange. All the participants are therefore able to reconstruct all the items.

- *Step 5 (message recovery):* the parties who have not received expected messages in the previous steps contact the other parties to obtain the necessary shares to reconstruct the shared secrets s_i.

If we assume that all participants correctly follow the steps from 1 to 4, the complexity in terms of exchanged messages is linear to the number of participants when using a broadcasted channel, but becomes quadratic when using a non-broadcasted channel. The average size of the messages is also linear to the number of participants (either broadcasted or non-broadcasted).

One could point out that the protocol is suitable for one-to-many exchanges but that it cannot be used in one-to-one or many-to-many exchanges: this is due to the fact that all the parties in the environment have knowledge (or can have knowledge) of the exchanged items. A more important problem is that participants are not able to determine if the shares which are sent to them effectively allow to reconstruct the entire information. It is a fundamental point of fair exchange, and not to guarantee this property here renders the protocol null and void in practice.

In Chapter 4, we propose an optimistic two-party fair exchange protocol that is also based on a secret sharing scheme. Our protocol does not have these drawbacks.

Guardian Angels Model

CHAPTER TWO

Almost all existing fair exchange protocols are based on the same assumptions on the behaviors of the parties: the main participants are assumed to be (potentially) fully dishonest while third parties are always fully trusted. One exception is Franklin and Tsudik's protocol [81] that relies on a semi-trusted third party, as explained in Chapter 1. Another exception is Vogt, Pagnia, and Gärtner's protocol [173, 174], devoted to electronic commerce, which relies on stronger assumptions regarding the trust of the main participants. Indeed, a trusted security module is added to one participant: the customer. Any kind of item (even time-sensitive) can be exchanged, but only against a payment. The protocol involves four entities: the client, his security module, the vendor (which does not have security module), and the vendor's bank (which is only called in case of a conflict). The sketch of the protocol is the following: firstly, the client sends the payment and the expected item's description to his security module, and the vendor sends the item and its description also to the client's security module. After checking the item and the payment, the security module then sends the payment to the vendor. Finally, if the payment is correct, the vendor must send a payment acknowledgment to the security module which then gives the expected item to the client. If the vendor does not send the acknowledgment, the bank is called in order to restore the fairness since the vendor already has the payment. Thus [173, 174] falls into the optimistic fair exchange protocols category since a trusted third party is needed in case of conflict, despite the presence of security modules.

In this chapter, we propose a symmetric model where both main participants host a security module that is tamper-proof, so-called a *Guardian Angel* [28, 29]. In this model, we supply a fair exchange protocol suited to arbitrary kinds of items and that requires no central third party. Instead, the fairness relies on the virtually distributed trusted third party formalized by the Guardian Angels. Such a model is well suited to mobile ad hoc networks where having a centralized party is unrealistic and where each mobile may possess such a security module for other security features such as confidentiality of the communication.

Before describing our fair exchange protocol, we introduce the synchronization problem in which honest parties need to decide whether or not a protocol succeeded through a hostile network that does not guaranty liveliness. This problem can be investigated independently in order to add reliability of protocol termination in secure channels. We address this problem by providing a gradual synchronization protocol, called the *Keep-in-Touch* protocol (KiT). Then, we show in Section 2.2 that the fair exchange problem can be reduced to the synchronization problem in this model and design a probabilistic fair exchange protocol, based on the KiT protocol, which provides arbitrarily low unfairness. The protocol requires neither trusted third party nor computational power assumptions. In this framework, it is the first protocol that takes advantage of the presence of security modules. We finally conclude this chapter by giving an application of our protocol in ad hoc mobile networks.

2.1 Synchronization

2.1.1 Secure Communications over Insecure Channels

We saw in Section 1.3.1 that packet security consists of confidentiality, authentication, integrity, liveliness, and timeliness, while session security consists of sequentiality and safe termination. Once packet authentication and integrity is protected, e.g., using MAC, sequentiality is typically assured by using a packet sequence number. The (maybe implicit) sequence number is authenticated together with the packet. This is the case in TLS 1.0 [63] and SSH [182]. Authenticated modes of operation for block ciphers such as CCM [179] also provide such a security feature. Hence, provided that one can securely establish symmetric key materials, it is quite easy to achieve authentication, integrity, confidentiality, and sequentiality of a protocol session. However, if liveliness is not guaranteed on the packet level, the safe termination of the protocol can be compromised: one party can terminate a session while the other ends up in an unresolved state. This lack of safe termination can be quite problematic in the case of two remote participants willing to perform an electronic transaction: one participant may think that the transaction has succeeded while the other doubts whether the transaction is considered as successful or not. This problem occurs in many well-known tunneling techniques, for example TLS 1.0 [63] and SSH [182].

In TLS, the client and the server must close connections during the execution of a session. In order to do this, “the client and the server must share knowledge that the connection is ending in order to avoid a truncation attack” [63]. The standard suggests a simple procedure which consists of sending a `close_notify` message from the initiator to the other party. This prevents new connections from opening in the session until the `close_notify` is properly acknowledged. If the session continues with new connections, it means that the previous connections had closed safely. However, this scheme obviously lacks safe termination, at least for the very last connection.

In SSH, when either party wishes to terminate the channel, it sends to the other party a message `ssh_msg_channel_close`. Upon receiving this message, a party must reply by sending back the same message. [182] specifies that “the channel is considered closed for a party when it has both sent and received `ssh_msg_channel_close`”. However, it does not specify what happens when this message is not received by the originator of the termination.

It only notes that "[SSH] is designed to be used over a reliable transport. If transmission errors or message manipulations occur, the connection is closed [...]. Denial of service attacks of this type ('wire cutter') are almost impossible to avoid".

We address this problem below.

2.1.2 Towards a Probabilistic Synchronization Protocol

We assume that two participants can communicate over a channel that protects packet authentication, integrity, and confidentiality, as well as session sequentiality. They would like to mutually agree on when a session is completed. We formalize the *synchronization* problem below.

Definition 3. *A synchronization protocol specifies two communicating algorithms A and B that communicate over a secure channel that may be subject to malicious disconnection. Both algorithms start with an input bit and terminate with an output bit. We require that*

- *A and B eventually halt;*
- *no algorithm yield 1 if its input is 0;*
- *when the channel is not disconnected, A and B always yield the product of the two input bits.*

The synchronization succeeds if both A and B terminate with the same output.

Even and Yacobi [71] proved the impossibility of designing a two-party synchronization protocol that always succeeds. We bypass this difficulty by addressing the synchronization problem in a probabilistic way. We define two quality measures:

- P_a (probability of asymmetric termination) is the maximum probability that the protocol fails, over all possible misbehavior strategies.
- P_c (probability that the crime pays off) is the maximum conditional probability that the protocol fails, conditioned on the protocol being interrupted by a malicious disconnection, over all possible misbehavior strategies.

Note that there is a tricky distinction between P_a and P_c which will be shown in the sequel. P_a gives confidence to A and B that the protocol will succeed while P_c measures the incentive for misbehavior. We recall that the communication is secure except the safe termination, and therefore the only way to attack the synchronization protocol is to disconnect A and B at some point during the session.

2.1.3 The Coordinated Attack Problem

The synchronization problem defined in Section 2.1.2 was first described in the domain of distributed algorithms by Gray [97], where it was called the *generals paradox.* Since then, it has usually been called the *coordinated attack problem.* We give below the description of the coordinated problem as suggested by Lynch in [125]:

> *"Several generals are planning a coordinated attack from different directions, against a common objective. They know that the only way the attack can succeed is if all the generals attack; if only some of the generals attack, their armies will be destroyed. Each general has an initial opinion about whether his army is ready to attack. The generals are located in different places. Nearby generals can communicate, but only via messengers that travel on foot. However, messengers can be lost or captured, and their messages may thus be lost. Using only this unreliable means of communication, the generals must manage to agree on whether or not to attack. Moreover, they should attack if possible."*

Varghese and Lynch analyzed in [170, 171] the *randomized* coordinated attack problem, and shown that any r-round protocol for this problem has probability of disagreement at least $\frac{1}{r+1}$. The Keep-in-Touch protocol proposed in the next section is rather similar to the randomized protocol proposed in [170] when two generals are present.

2.1.4 Keep-in-Touch Protocol

The Keep-in-Touch (KiT) protocol is a synchronization protocol whose principle is quite simple: if A's input is 1 then, it picks a random number $C \geq 0$. C represents the number of messages that should be exchanged after B joins the protocol by sending a first empty message. Then, if both inputs are 1, A and B just "keep in touch" by sequentially exchanging authenticated empty messages. One can notice that these C exchanged messages are actually empty ones! In case a time-out occurs while expecting a message, a participant stops and yields 0.

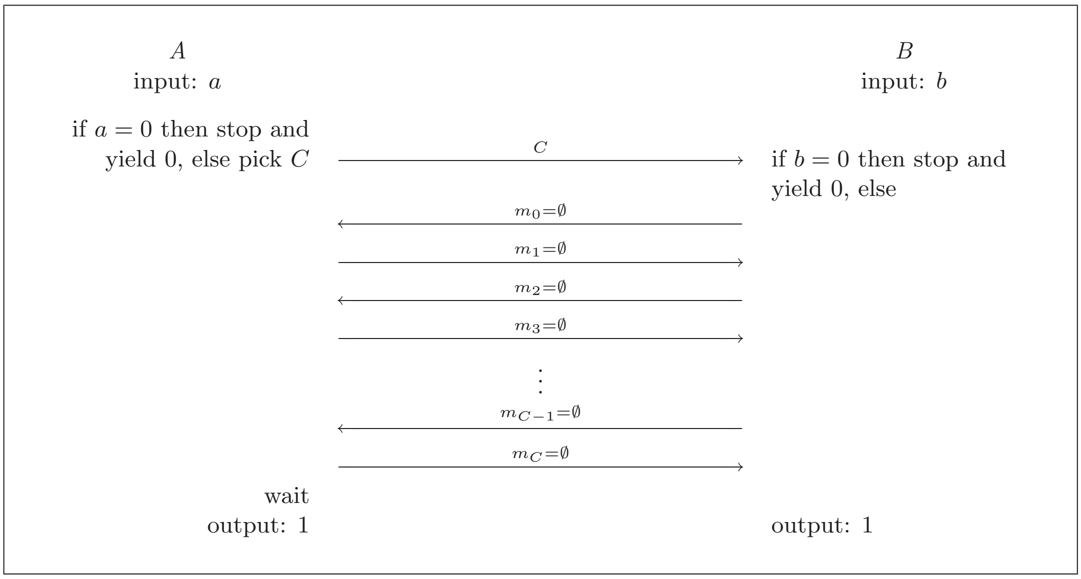


Figure 2.1: Keep-in-Touch (KiT) protocol

Termination Side Channel Protection

In the case where the adversary has access to the output of A or B through a side channel, the last sender should wait for a given period larger than the time-out before terminating. This ensures that both A and B complete before the adversary gets any side information. The last receiver could still acknowledge the last message to prevent the other party from waiting, but disconnection at this point should not change the output. The consequence of such an attack is only a time loss for the waiting participant.

Timeout Removal

Similarly, when $a = 0$, A can prevent B from waiting by sending a specific message. The case in which $a = 1$ and $b = 0$ is similar.

Complexity

When no attack occurs, the complexity in terms of exchanged messages is exactly equal to $C + 2$. When the channel is cut, the complexity is smaller, so we can just focus on C. We let p_i be the probability $\Pr[C = i]$. By definition, the average complexity is $2 + E(C)$, where

$$E(C) = \sum_{i=0}^{+\infty} ip_i.$$

Note that the communication and time complexities are linear in terms of C due to the simplicity of the message contents and the computations to perform.

Synchronization

Obviously, all properties of Definition 3 are satisfied, hence we have a synchronization protocol.

Quality Measures

Clearly, disconnecting A from B in the first message makes A and B output 0 and the protocol succeeds. We now assume that the adversary is willing to drop message m_i. If $C < i$ then the attack has no influence and the protocol successfully terminates. If $C > i$, the participant who is expecting m_i cannot send the next one, so both participants are blocked and the protocol safely succeeds since both A and B yield 0 after timeouts expire. Clearly the protocol fails if $C = i$, that occurs with probability p_i. Therefore we have

$$P_a = \max_i p_i.$$

With the same discussion we can show that the above misbehavior has a conditional probability of success of $\Pr[C = i | C \geq i]$. Hence we have

$$P_c = \max_i \frac{p_i}{\sum_{j \geq i} p_j}.$$

Theorem 1. *The KiT protocol is a synchronization protocol. Let $p_0, p_1, \ldots$ denote the probability distribution of C in the protocol. The expected complexity is $2 + E(C)$ where $E(C) = \sum_i ip_i$, the probability of asymmetric termination is $P_a = \max_i p_i$ and the probability that the crime pays off is $P_c = \max_i p_i / \sum_{j \geq i} p_j$.*

Example 1. *For any n, when $p_0 = \cdots = p_{n-1} = \frac{1}{n}$ and $p_i = 0$ for $i \geq n$ we have $E(C) = \frac{n-1}{2}$ and a probability of asymmetric termination of $P_a = \frac{1}{n}$. However we have $P_c = 1$ for $i = n-1$. In other words, if the strategy of the adversary is to disconnect at m_{n-1} then his risk is void since this is definitely the last message.*

Example 2. *For any p, when $p_i = (1-p)^i p$ for $i \geq 0$ we have $E(C) = \frac{1}{p} - 1$ and a probability of asymmetric termination of $P_a = p$. In this case we also have $P_c = p$.*

Optimal Distributions for the KiT Protocol

The distribution choice plays on the complexity and the parameters P_a and P_c. Obviously there is a trade-off. The optimal case is studied in the following theorem.

Theorem 2. *Let $p_0, p_1, \ldots$ denote the probability distribution of C in the KiT protocol. We have $E(C) \geq \frac{1}{2}\left(\frac{1}{P_a} - 1\right)$ and $E(C) \geq \frac{1}{P_c} - 1$ where P_a and P_c are the probability of asymmetric termination and the probability that the crime pays off respectively.*

This theorem shows that Example 1 is the optimal case for P_a and that Example 2 is the optimal case for P_c.

Proof. We want to minimize $E(C)$ for a given P_a. It is equivalent to finding $p_0, p_1, \ldots$ such that $0 \leq p_i \leq P_a$ for all i, $\sum p_i = 1$, and $\sum ip_i$ minimal. Let $n = \lfloor \frac{1}{P_a} \rfloor$ and $\alpha = \frac{1}{P_a} - n$. We have $\alpha \in [0, 1[$. Obviously $\sum ip_i$ is minimal when the first p_is are maximal, i.e., when $p_0 = p_1 = \cdots = p_{n-1} = P_a$. The sum of all remaining p_i is equal to $1 - nP_a$. Thus we have

$$E(C) \geq P_a + 2P_a + \cdots + (n-1)P_a + n(1 - nP_a).$$

Hence $E(C) \geq \frac{n(n-1)}{2}P_a + n(1 - nP_a)$. If we substitute $\frac{1}{P_a} - \alpha$ to n we obtain

$$E(C) \geq \frac{1}{2}\left(\frac{1}{P_a} - 1\right) + \frac{\alpha P_a}{2}(1 - \alpha).$$

Since $0 \leq \alpha < 1$ we have $E(C) \geq \frac{1}{2}\left(\frac{1}{P_a} - 1\right)$. This proves the first bound.

For the second bound we notice that

$$E(C) = \sum_{i=1}^{+\infty} \sum_{j \geq i} p_j = \sum_{i=0}^{+\infty} \sum_{j \geq i} p_j - 1.$$

Since we have $\sum_{j \geq i} p_j \geq \frac{p_i}{P_c}$ for all i by definition of P_c, we obtain that $E(C) \geq \frac{1}{P_c} - 1$. □

Bit-Messages Variant

Instead of picking C once and sending it at the beginning of the protocol, we can just ask each participant to toss a biased coin before sending m_i and sending the output bit in the message. A 0 bit means "let's keep in touch" and a 1 bit means "so long". Obviously, if the ith bit is 1 with probability $\Pr[m_i = 1] = \Pr[C = i | C \geq i]$, this variant is fully equivalent to the above protocol. Example 2 is equivalent to $\Pr[m_i = 1] = p$ for all i.

2.1.5 Optimality of the KiT Protocol

We prove in this section that our protocol is the most efficient one within our settings. We also show that perfect synchronization cannot be assured with probability 1 with a finite complexity.

Theorem 3. *For any synchronization protocol between two participants A and B (initiated by A) with parameters P_a and P_c, we let $C + 2$ denote the number of exchanged messages. We can define a random variable C' such that $\Pr[C' \leq C] = 1$ and that C' defines a KiT protocol $A'B'$ with parameters P'_a and P'_c such that $P_a \geq P'_a$ and $P_c \geq P'_c$.*

Proof. We are given two (probabilistic) algorithms A and B which exchange $C + 2$ messages under normal conditions. Without loss of generality, we assume that A initiates the protocol. We now construct a KiT protocol $A'B'$ which uses $C' + 2$ messages. Note that we only need to define how A' computes C' since the remaining part of A' and B' are fully specified by the KiT protocol.

We first note that when either input is 0, the KiT protocol is always optimal: sending less number of messages may lead to cases that would violate the definition of the synchronization protocol. We deduce that C must be positive when both inputs are 1. We concentrate on this case in what follows.

In order to compute C', A' first simulates a normal interaction between A and B with input 1. We assume that all random coins are set in advance by A' so that the simulation is run on deterministic algorithms. Note that the simulator can freely restart A or B in a previous state. Hence, A' can define the following quantities based on a random instance of the simulation. Let $x_1, x_2, \ldots, x_{C+2}$ be a sequence of bits in which x_i is equal to 0 when the ith message is sent from A to B, and to 1 when it is sent in the other direction. By definition of the synchronization protocol, both A and B yield 1 after the final message. A' now analyzes the final output in case the adversary disconnects the channel at the ith message (i.e., this message is sent but never received) for $i = 1, \ldots, C + 2$. We define a_i (resp. b_i) as the final output of A (resp. B) if the channel is disconnected at the ith message. We let $C' + 2$ be the smallest i such that $a_j = b_j = 1$ for any $j \geq i$. Obviously we always have $C' \leq C$. In the rest of the proof we demonstrate that the P'_a and P'_c parameters for the $A'B'$ protocol are no larger than the P_a and P_c parameters for the AB protocol. In order to do this we show that any attack strategy S' against $A'B'$ can be transformed into an attack strategy S against AB with at least the same probability of success.

An attack S' against $A'B'$ is fully defined by the index i of the message from which the channel is disconnected. We consider the attack S against AB which cuts the channel when the ith message is sent. The attack S' succeeds only for instances of the AB simulation in which $i = C' + 2$. Below we show that S also succeeds for the same instances. We deduce that no attack against $A'B'$ is more successful than any attack against AB. Hence $P_a \geq P'_a$ and $P_c \geq P'_c$.

Let us assume that $x_i = 0$. By definition of C' we have $a_{C'+2} = b_{C'+2} = 1$, but we do not have $a_{C'+1} = b_{C'+1} = 1$. We notice that $a_{C'+1} = a_{C'+2}$ since everything is normal for A. Thus we have $b_{C'+1} = 0$. Since B does not receive the ith message, it eventually yields 0 while A yields 1. A similar argument holds for $x_i = 1$. □

2.2 Fair Exchange with Guardian Angels

2.2.1 Pirates and Guardian Angels

The Guardian Angels model, depicted in Figure 2.2, is based on the "observer" notion introduced by Chaum and Pedersen [54] in 1992. It considers that both participants own a security module. Contrary to [54] we assume that the security modules are honest. For this reason, participants and security modules are called "Pirates" and "Guardian Angels" respectively.

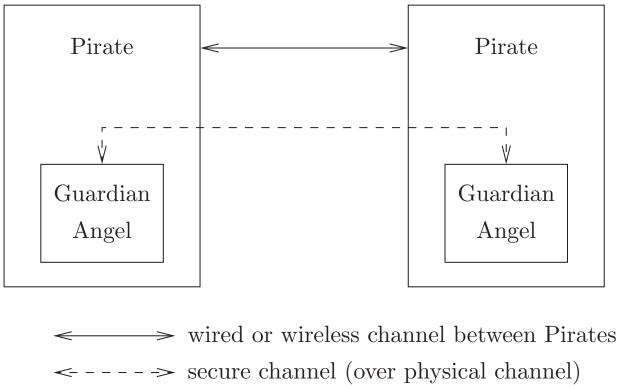

Pirate
Pirate
Guardian Angel
Guardian Angel
wired or wireless channel between Pirates
secure channel (over physical channel)

Figure 2.2: Pirates and guardian angels

Pirates are powerful in the sense that they are able to communicate with all other devices and their own Guardian Angel. We require no assumptions on the computational capabilities of Pirates, since they can vary a lot. We have no assumptions on the inter-Pirates communication channels. While on one hand they can be totally insecure, on the other hand the Pirate-Guardian Angel communication channel is assumed to be fully secure: it provides confidentiality, integrity, authentication, sequentiality, and timeliness.

Guardian Angels fulfill the following requirements: they are *tamper-proof*, that is any potential adversary could not have access to the stored data or change the Guardian Angel's behavior. Full access stays possible but limited to certain authorized parties, e.g., for set up. Since the cost of making a tamper-proof device increases steadily with its capabilities, we assume that Guardian Angels are simple and limited devices: their computational and storage capabilities are low. Moreover they have a single I/O port that is only connected to their own Pirate: they have no other information source about the outside world. In particular, we assume that they have no notion of time but for a clock signal that is provided by the Pirate.

Recently, manufacturers have begun to equip hardware with such modules: for instance smart cards or special microprocessors. Examples include the "Embedded Security Subsystem" within the recent IBM Thinkpad or the IBM 4758 secure co-processor board [66]. In fact, a large body of computer and device manufacturers have founded the Trusted Computing Group [168] to promote this idea. Because their hardware is tamper-proof, the software running within the security modules is certified. Thus the modules can communicate through secure channels. In certain settings, the overall system can even assumed to be synchronous, i.e., it is reasonable to assume an upper bound on the relative speeds of honest parties (and their security modules) as well as on the communication delays between them.

Although Guardian Angels can establish secure channels among them, providing confidentiality, integrity, authentication, and sequentiality, those channels require the cooperation of Pirates. Consequently, timeliness cannot be guaranteed and the inter-Guardian Angels communication channels correspond to the model given in Section 2.1. Results of this latter section are therefore used to design a probabilistic fair exchange protocol.

2.2.2 Fair Exchange with Two Guardian Angels

Let us denote P the Pirates and G the Guardian Angels. In this section we focus on the fair exchange problem between P_A and P_B using G_A and G_B.

If P_A and P_B wish to exchange m_A and m_B, they ask their Guardian Angels for assistance. Then, G_A simply sends m_A to G_B through their secure channel, and G_B transmits m_B to G_A. After the exchange itself, they perform a synchronization by using the KiT protocol. If the protocol succeeds, G_A and G_B disclose the received items to the Pirates. This protocol is illustrated in Figure 2.3.

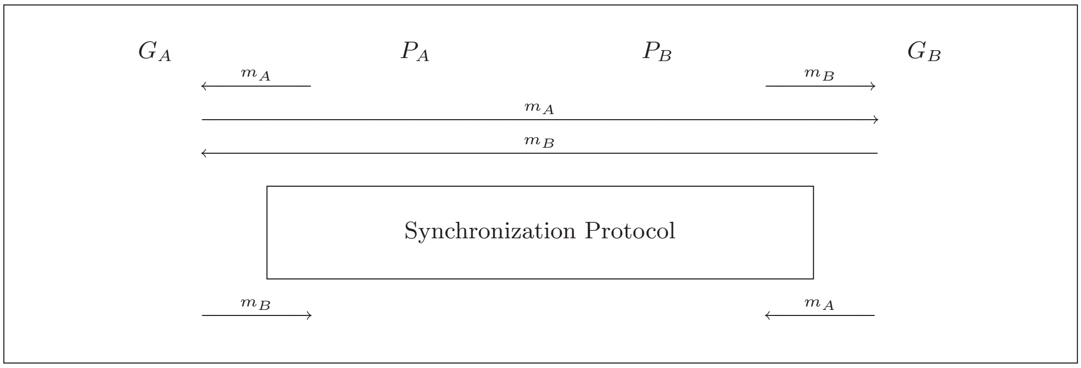


Figure 2.3: Fair exchange with two guardian angels

To keep the protocol easily readable, certain important issues are not depicted in the figure. Firstly, we assume that the Guardian Angels have means to check that the received items are the expected ones: Pirates can send the descriptions of the items to their Guardian Angels. Secondly, since items have an arbitrary size and that Guardian Angels are assumed to have a limited storage facility, we assume that the Guardian Angels forward the received item by encrypting it on-the-fly with a freshly picked key. The key then replaces the item in the above protocol. Thirdly, the lack of timeliness in the synchronization should be managed by timeouts. Because Guardian Angels are not assumed to have internal clocks, timeouts should be yield by Pirates. Similarly, the end of the synchronization protocol (at least for the

Guardian Angel who sends the very last message) should occur only after the Pirate yields a timeout.

Obviously, the fair exchange protocol inherits from the properties of the KiT protocol. Quality measures P_a and P_c given in Theorem 1 can be interpreted respectively as the probability for an instance of the protocol to be unfair and the probability that cutting the channel at some point pays off.

Below we give the initiator's fair exchange programs.

```
FairExchangeGuardianInitiator
  receive $m_A$ from $P_A$
  establish a secure channel with $G_B$ through $P_A$
  send $m_A$ to $G_B$ through the channel
  receive $m_B$ from $G_B$ through the channel
  check $m_B$, if incorrect then abort endif
  encrypt $m_B$ with a random secret key $K$
  send $m_B$ encrypted with $K$ to $P_A$
  execute the synchronization protocol with input 1
  if the synchronization outputs 0 then abort endif
  send $K$ to $P_A$
```

Figure 2.4: Fair exchange program of the guardian angel G_A

```
FairExchangePirateInitiator
  send $m_A$ to $G_A$
  forward messages between $G_A$ and $P_B$ for the channels between $G_A$ and $G_B$
  if timeout then
    send timeout signal to $G_A$
    if receive $K$ then decrypt $m_B$ else the protocol failed endif
  endif
```

Figure 2.5: Fair exchange program of the pirate P_A

2.3 Applying to Mobile Ad Hoc Networks

Current mobile networks rely on a heavy fixed infrastructure that connects users through relays. Installing such an infrastructure is often either too expensive or technically impossible and hence some areas are not reachable. Mobile ad hoc networks mitigate this problem by allowing users to route data through intermediate nodes: the network is furthermore self-organized and no more relies on any established infrastructure.

In such a network, cryptographic protocols cannot use on-line trusted third party for some obvious reasons. One may think that optimistic protocol could run properly. However, using off-line trusted third party does not come up to the mobile ad hoc networks requirements since we cannot assume that most of the nodes will be honest. Indeed, the nodes have to forward other's packets to keep the community alive, but they will try to cheat as soon as possible in order to save their battery life since forwarding packets have a substantial cost: nodes will become selfish. Participants will then require the trusted third party in each transaction. On the other hand we cannot reasonably use gradual fair exchange protocols since no assumptions have been made on the computational power of the nodes: they could be mobile phones, but also PDAs, laptops, etc.

Additionally, extreme cases of fully self-organized networks aim at getting rid of any central service. Our model is then fully relevant to this environment since it achieves probabilistic fairness without trusted third party, assuming only that a secure channel between the Guardian Angels is available. Even if economic and practical aspects are not fully designed yet, it makes sense to imagine the following scenario. Some company builds and sells Guardian Angels who become the virtual distributed trusted third party. This company (who replaces the network provider of earlier mobile networks) simply makes community rules and installs an accounting service. Guardian Angels are responsible for enforcement of community rules and keep local accounting in their lifetime. When they expire, their successors keep track of the accountings. Thus, users have only to buy and plug Guardian Angels into their device in order to control fairness, security, accounting, services, etc. We can later imagine several Guardian Angels manufacturers with specific trade exchange protocols.

As an application we can exchange an inexpensive service against a micropayment, e.g., incentives for routing [17, 110], with the protocol of Example 2. When $p = 1/2$, the risk for both participants is to loose small valuables, but with a probability bounded by $1/2$ instead of a probability which may be equal to 1.

Gracefully Degrading Multi-Party Fair Exchange

CHAPTER THREE

We have shown in Chapter 2 how a two-party fair exchange can be solved probabilistically by using guardian angels. We now use a similar model in order to solve multi-party fair exchange. Below, we provide a synchronous distributed algorithm [20, 22] aimed at exchanging digital items among untrusted parties, each hosting a guardian angel. As an underlying building block, our algorithm uses an *early stopping* subprotocol that solves a specific variant of consensus – we call *biased consensus* – in the *general omission failure model.* The algorithm provides the following two complementary features:

- If a majority of the parties are honest, then the algorithm deterministically guarantees the completeness, fairness, and termination properties. This is optimal in terms of resilience: we indeed show that, even in a synchronous model with guardian angels, no deterministic algorithm solves fair exchange if half of the parties are dishonest.
- If at least half of the parties turn out to be dishonest, then our algorithm degrades gracefully in the following sense: it still guarantees the completeness and termination properties, as well as ensures that the probability of violating fairness can be made arbitrarily low.

Section 3.1 gives an intuitive idea of the protocol. Section 3.2 extends the guardian angels model introduced in Chapter 2 to the multi-party case. Section 3.3 introduces biased consensus, and shows the equivalence between fair exchange and biased consensus in the guardian angels model. The impossibility of deterministic multi-party fair exchange without a honest majority motivates our notion of *gracefully degrading fair exchange* (GDFE), also introduced in Section 3.3. Section 3.4 describes our gracefully degrading fair exchange algorithm and states its optimality. Thus, the probability distribution that optimizes the average communication complexity is given. We finally show that it is inversely proportional to the probability of violating fairness.

3.1 Sketch of the Protocol

Our algorithm is made of three phases. Below, we provide the intuition behind each phase.

1. In the first phase, called the *initialization* phase, the guardian angels exchange the items that are supposed to be traded by the pirates. These items are not delivered by the guardian angels to their pirates at this stage: this is only performed if the third phase (described below) successfully terminates. Any guardian angel can decide to abort the exchange at this time if some item is missing or does not match its expected description. The guardian angel that is hosted by the party that initiates the exchange additionally selects a random number C. This C is disseminated among all other guardian angels. The role of this random number is crucial in the second phase of the algorithm.

2. In the second phase, called the *fake* phase, all the guardian angels exchange messages during C rounds; each round following the same communication pattern as in the third phase (below). The fact that the random number C, determined in the first phase, is not accessible to the pirates is fundamental here. Roughly speaking, the goal of this phase is to make the probability, for any number of dishonest parties to successfully guess when the actual agreement phase will take place (third phase below), arbitrarily low. If any dishonest party drops a message towards a honest party in this phase, the guardian angel hosted by the latter simply aborts the exchange and forces other guardian angels to abort the protocol as well, thus penalizing any dishonest pirate that might try to bias the exchange in its favor.

3. In the third phase, called the *agreement* phase, the guardian angels solve a problem we call *biased consensus*. In this problem, the processes (in our case the guardian angels) start from an initial binary value (a proposal) and need to decide on a final binary value: either to abort the exchange or commit it (and deliver the items to their pirates). Unlike in consensus [79], but like in non-blocking atomic commit (NBAC) [37, 161], the problem is biased towards 1: no process can decide 1 if some process proposes 0. The agreement aspect of this problem is however different from consensus and NBAC; it is also sometimes biased towards 0: we simply require here that, if some process decides 1, then no correct process decides 0. We consider an early stopping algorithm that solves this problem in the general omission failure model, along the lines of [147].

3.2 Extension of the Guardian Angels Model

3.2.1 Pirates and Guardian Angels

The system we consider is composed of a set of processes, some modeling pirates and the other modeling guardian angels. These processes communicate by exchanging messages. Two processes connected by a physical channel are said to be adjacent. We assume that there exists a fully connected communication topology between the pirates, i.e., any two pirates are adjacent. Furthermore, we assume that every pirate P_A is adjacent to exactly one guardian angel G_A (i.e., there is a bijective mapping between guardian angels and pirates): we say that P_A is *associated* with G_A. No two guardian angels are adjacent. In other words, for any two guardian angels G_A and G_B to communicate, they need to do so through P_A and P_B. This indirection provides the abstraction of an overlay network at the guardian angels level. We call the part of the system consisting of guardian angels, and the virtual communication links between them the *secure subsystem*. We call the part of the system consisting of pirates and the communication links between them the *untrusted system*. The notion of association can be extended to systems, meaning that, for any given untrusted system, the *associated secure subsystem* is the system consisting of all guardian angels associated to any pirate in that untrusted system.

Recently, Benenson, Gärtner, and Kesdogan [35] have also used this model in order to improve secure multi-party computation.

3.2.2 Guardian Angels and Virtual Channels

Guardian angels are interconnected by a virtual communication network with bidirectional channels over the physical communication network among the pirates. For simplicity, we denote the guardian angels by $G_1, \ldots, G_n$. We assume that between any two guardian angels G_i and G_j, confidentiality, authentication, and integrity are guaranteed. Furthermore, when no attack occurs, a message sent by G_i to G_j is always delivered at G_j within some known bound Δ on the waiting time.

3.2.3 Trust and Adversary Model

Guardian angels can be trusted by other guardian angels or pirates, but pirates cannot be trusted by anybody. Pirates may be malicious, i.e., they may actively try to fool a protocol by not sending any message, sending wrong messages, or even sending the right messages at the wrong time. However, we assume that pirates are computationally bounded. In particular, brute force attacks on secure channels are not possible. Guardian angels are supposed to be cheap devices without their own source of power. They rely on power supplied by their pirates. A Pirate may inhibit all communication between its associated guardian angel and the outside world, yielding a channel in which messages can be lost.

A pirate *misbehaves* if it does not correctly follow the prescribed algorithm and we say that it is *dishonest*. Otherwise it is said to be *honest*. Misbehavior is unrestricted (but computationally bounded as we pointed out). Guardian angels always follow their protocol, but since their associated pirates can inhibit all communication, this results in a system of guardian angels with unreliable channels (the general omission failure model [143], i.e., where messages may not be sent or received). In such systems, misbehavior (i.e., failing to send or receive a message) is sometimes termed *failure*. We call guardian angels associated with honest pirates *correct*, whereas those associated with dishonest pirates *faulty*. In a set of n pirates, we use t to denote a bound on the number of pirates which are allowed to misbehave and f the number of pirates which actually do misbehave ($f \leq t$). Sometimes we restrict our attention to the case where $t < n/2$, i.e., when a majority of pirates are assumed to be honest. We call this the *honest/correct majority assumption*.

Our adversary model is based on the strongest possible attack, the case in which all of the f dishonest pirates collude. We assume that the adversary knows all the algorithms and the probability distributions that are used.

3.3 From Biased Consensus to Fair Exchange

In this section we show that fair exchange (FE), at the pirates level, is in a precise sense equivalent to a problem that we call *biased consensus* (BC), at the level of the underlying guardian angels. Then, we state that biased consensus is impossible if half of the processes can be faulty and derive the impossibility of fair exchange if a majority of pirates can be dishonest. This motivates our definition of a weaker variant of fair exchange, called gracefully degrading fair exchange.

3.3.1 Biased Consensus

Consider the following variant of consensus in a model where processes can fail by general omissions [143].

Definition 4 (biased consensus). *A protocol solves biased consensus (BC) if it satisfies:*

- Termination: *every correct process eventually decides.*
- Non-triviality: *if no process is faulty and no process proposes 0, then no correct process decides 0.*
- Validity: *if any process proposes 0, then no process decides 1.*
- Biased agreement: *if any process decides 1, then no correct process decides 0.*

Processes invoke biased consensus using primitive BCpropose(v), v being the vote of the process, a binary value either 0 or 1. Possible decisions are 0 (*abort*) and 1 (*commit*). Vote and decision correspond to input and output defined in the KiT protocol (Section 2.1). Termination, non-triviality, and validity are the same as in NBAC, whereas the biased agreement is weaker than the agreement property of NBAC [161].

Below, we show that FE and BC are equivalent in our model.

Theorem 4. *For any n and t, biased consensus is solvable in the secure subsystem, if and only if fair exchange is solvable in the associated untrusted system.*

Proof. Assume that we have a solution to BC in the secure subsystem consisting of guardian angels $G_1, \ldots, G_n$. Now consider the algorithm depicted in Figure 3.1. This is a wrapper around the BC solution that solves FE at the pirates level. In other words, it is a reduction of FE into BC in our model. In the algorithm, a pirate hands to the associated guardian angel its item and the executable description of the desired item, as well as identifiers of the pirates with which items should be exchanged. The guardian angel exchanges the item with its partners, and then checks the received item (initialization phase). Finally all guardian angels agree on the outcome using BC (agreement phase). The proposal value for BC is 1 if the check was successful and no abort was requested by the pirate in the meantime. If

```
FairExchange($myitem$, $description$, $source$, $destination$)
    ⟨send $myitem$ to $destination$ over secure channel⟩
    **timed wait for** ⟨expected item $i$ from $source$ over secure channel⟩
    ⟨check $description$ on $i$⟩
    **if** ⟨check succeeds and no timeout⟩ **then** $v := 1$ **else** $v := 0$ **endif**
    $result :=$ BCpropose($v$)
    **if** $result = 1$ **then** return $i$ **else** return ⟨abort⟩ **endif**
```

Figure 3.1: Using biased consensus to implement fair exchange: code of the guardian angel

BC terminates with the process deciding 1, then the guardian angel releases the item to the pirate. We now discuss each property of fair exchange.

The termination property of FE is guaranteed because BC assures termination. Consider completeness and assume that all participating pirates are honest and all items match their descriptions. All votes for BC will thus be 1.

Now the non-triviality property of BC guarantees that all processes (recall that every process is correct) will decide 1 and subsequently return the item to their pirates.

Consider now fairness and observe that, in our adversary model, no dishonest pirate can derive any useful information from merely observing exchanged messages over the secure channels. The only way to receive information is through the interface of the FairExchange procedure. If one item does not match the description at some process, then this process will engage in BC with a $v = 0$ (note that this will happen even if the associated pirate is dishonest).

Validity of BC implies that the exchange results in no process deciding 1, so none of the pirates receives anything from the exchange. Additionally, if some honest pirate receives nothing through the exchange, then the biased agreement property of BC implies that no pirate can receive anything.

Conversely, biased consensus can be implemented using fair exchange by invoking the procedure BCpropose(v_i) = FairExchange$(v_i, 1, G_{i-1}, G_{i+1})$ at every process G_i $(1 \leq i \leq n)$, with the convention that $G_0 = G_n$ and $G_1 = G_{n+1}$. Remember that 1 is the description of the expected item v_i, which corresponds to the vote in the biased consensus. Here we assume that if FE returns ⟨*abort*⟩ instead of the item, BC returns 0. So the votes, being exchange items in this case, are exchanged in a circular fashion among guardian angels. In other words, we use a FE protocol with a ring topology (see Section 1.2.3). It is not difficult to see that the FE properties guarantee the BC properties. This is immediate for termination and non-triviality. Consider now validity and assume that G_j proposes 0 to BC. The item description checking at G_{j+1} will fail and the first point of Definition 2 guarantees that every process that decides in BC decides 0. The second part of fairness guarantees a biased agreement. □

Theorem 5. *Consider a synchronous system where processes can fail by general omissions. No algorithm solves biased consensus if $\lceil \frac{n}{2} \rceil$ processes can be faulty.*

Proof. We divide the set of processes into two sets, S_1 and S_2, each containing at most $\lceil \frac{n}{2} \rceil$ processes. In a given run of BC algorithm, the k-round configuration $\mathcal{C}$ denotes the configuration of the system at the end of round k in that run. We define some runs that extend a k-round configuration as follows: $\mathsf{run}_1(\mathcal{C})$ is a run in which, after round k, the processes in S_2 fail in such a way that, in every round after round k and for each $i \in \{1, 2\}$:

- processes in S_i receive messages from other processes in S_i, and
- no process in S_i receives any message from any process in S_{3-i}.

Basically, the processes in S_2 fail by send omission while sending message to processes in S_1, and fail by receive omission when receiving message from processes in S_1. We define $\mathsf{run}_2(\mathcal{C})$ symmetrically: a run in which, after round k, the processes in S_1 fail in such a way that, in every round after round k and each $i \in \{1, 2\}$:

- the processes in S_i receive messages from other processes in S_i, and
- no process in S_i receives any message from any process in S_{3-i}.

Here processes in S_1 fail by send omission while sending message to processes in S_2, and fail by receive omission when receiving messages from processes in S_2. It is important to observe that no process can distinguish $\mathsf{run}_1(\mathcal{C})$ from $\mathsf{run}_2(\mathcal{C})$.

We denote by $\mathsf{val}_1(\mathcal{C})$ the decision value of processes in S_1 in $\mathsf{run}_1(\mathcal{C})$. Since all processes in S_1 are correct, all of them decide on the same value in $\mathsf{run}_1(\mathcal{C})$. Since no process in S_1 can distinguish $\mathsf{run}_1(\mathcal{C})$ from $\mathsf{run}_2(\mathcal{C})$, the processes in S_1 decide $\mathsf{val}_1(\mathcal{C})$ in $\mathsf{run}_2(\mathcal{C})$ as well. Similarly, $\mathsf{val}_2(\mathcal{C})$ denotes the decision value of processes in S_2 in $\mathsf{run}_2(\mathcal{C})$, and hence, in $\mathsf{run}_1(\mathcal{C})$ as well.

Suppose by contradiction that there is an algorithm A that solves BC when $\lceil \frac{n}{2} \rceil$ processes can fail. Consider a run r of A in which every process proposes 1 and no process is faulty. By the non-triviality property of BC, every process decides 1 in r, say at round z. Let us denote the system configuration at the end of round k in r by $\mathcal{C}_k$. We now try to determine $\mathsf{val}_1(\mathcal{C}_k)$ and $\mathsf{val}_2(\mathcal{C}_k)$.

We claim that $\mathsf{val}_1(\mathcal{C}_0)$ is 0. To see why, notice that the processes in S_1 are correct in $\mathsf{run}_1(\mathcal{C}_0)$ and they never receive any message from the processes in S_2. Thus, processes in S_1 cannot distinguish $\mathsf{run}_1(\mathcal{C}_0)$ from $\mathsf{run}_1(D)$, where D is an initial configuration in which processes in S_1 propose 1 and processes in S_2 propose 0. From the validity property, processes in S_1 decide 0 in $\mathsf{run}_1(D)$, and hence, in $\mathsf{run}_1(\mathcal{C}_0)$. Thus $\mathsf{val}_1(\mathcal{C}_0)$ is 0. Similarly, we can show that $\mathsf{val}_2(\mathcal{C}_0)$ is 0. Clearly, $\mathsf{val}_1(\mathcal{C}_z)$ is 1, because all processes decide (or rather has already decided) 1 in $\mathsf{run}_1(\mathcal{C}_z)$. Similarly, $\mathsf{val}_2(\mathcal{C}_z)$ is 1.

We now claim that for all rounds i such that $0 \leq i \leq z$, $\mathsf{val}_1(\mathcal{C}_i) = \mathsf{val}_2(\mathcal{C}_i)$. Assume that $\mathsf{val}_1(\mathcal{C}_i) = 0$ and $\mathsf{val}_2(\mathcal{C}_i) = 1$ (the contradiction for $\mathsf{val}_1(\mathcal{C}_i) = 1$ and $\mathsf{val}_2(\mathcal{C}_i) = 0$, is symmetric). Consider $\mathsf{run}_1(\mathcal{C}_i)$. The processes in S_1 are correct in $\mathsf{run}_1(\mathcal{C}_i)$ and they decide $\mathsf{val}_1(\mathcal{C}_i) = 0$. The processes in S_2 are faulty in $\mathsf{run}_1(\mathcal{C}_i)$, but they cannot distinguish $\mathsf{run}_1(\mathcal{C}_i)$ from $\mathsf{run}_2(\mathcal{C}_i)$ and hence, decide $\mathsf{val}_2(\mathcal{C}_i) = 1$ in $\mathsf{run}_1(\mathcal{C}_i)$. Thus $\mathsf{run}_1(\mathcal{C}_i)$ violates the biased agreement property.

Thus, for every round i such that $0 \leq i \leq z$, $\mathsf{val}_1(\mathcal{C}_i) = \mathsf{val}_2(\mathcal{C}_i)$. Thus there is a round j such that $\mathsf{val}_1(\mathcal{C}_j) = \mathsf{val}_2(\mathcal{C}_j) = 0$ and $\mathsf{val}_1(\mathcal{C}_{j+1}) = \mathsf{val}_2(\mathcal{C}_{j+1}) = 1$. Consider the following $j + 1$ round configuration $\mathcal{C}'$ that is obtained by extending $\mathcal{C}_j$ as follows: processes in S_2 fail (actually commit send omission) such that:

- for $i \in \{1, 2\}$, the processes in S_i receive messages from other processes in S_i,

- no process in S_1 receives any message from S_2 in round $j+1$, but
- the processes in S_2 receive the message from every process in S_1.

Observe that S_2 cannot distinguish $\mathcal{C}'$ from $\mathcal{C}_{j+1}$ and S_1 cannot distinguish $\mathcal{C}'$ from the $(j+1)$-round configuration of $\mathsf{run}_1(\mathcal{C}_j)$. Consider a run r' that extends $\mathcal{C}'$ in which S_2 omits to send and receive a message from S_1. Notice that S_1 is correct in r' and S_2 is faulty in r'. However, S_2 cannot distinguish r' from $\mathsf{run}_2(\mathcal{C}_{j+1})$, and hence, decides $\mathsf{val}_2(\mathcal{C}_{j+1}) = 1$ at the end of round z in r'. Furthermore, S_1 cannot distinguish r' from $\mathsf{run}_1(\mathcal{C}_j)$, and hence, decides $\mathsf{val}_1(\mathcal{C}_j) = 0$ at the end of round z in r'. Clearly, r' violates biased agreement. □

A direct corollary of Theorem 4 and Theorem 5 leads to the following result:

Theorem 6. *No algorithm solves fair exchange in our model if half of the pirates can be dishonest.*

3.3.2 Early Stopping Biased Consensus Algorithm

The BC algorithm we provide here (Figure 3.2, page 43) is an adaptation of the early stopping synchronous consensus algorithm of [147]. This algorithm solves BC if there is a majority of correct processes ($t < n/2$). It is early stopping in the sense that every process terminates in at most $\min(f+2, t+1)$ rounds. We recall that t denotes a bound on the number of pirates which are allowed to misbehave and f denotes the number of pirates which actually do misbehave ($f \leq t$). There are mainly two differences with the consensus algorithm of [147]: firstly the processes agree on a *vector* of initially proposed values rather then on a *set* of those (in the case of consensus); secondly we also introduce "dummy" messages to have a full information protocol [125], i.e., to have a uniform communication pattern in every round. In other words, in the original algorithm of [147] process G_i did not send a message to process G_j in a given round, while in our algorithm process G_i sends a dummy message m to process G_j, but process G_j disregards m. Our solution assumes all BC protocol messages to have the same size. The motivation for having a full information protocol and uniform message size, will be explained in Section 3.4.

The changes we make to the algorithm of [147] do not affect the correctness of the algorithm. The difference is that we introduce the procedure $\mathsf{decide}(\mathbf{v})$ (Figure 3.2), where $\mathbf{v}$ is a vector of initially proposed values processes agree on. Basically, every process G_i stores information about the value initially proposed by some process G_j at $\mathbf{v}_i[j]$. If the process G_i does not know the value that process G_j proposed, then $\mathbf{v}_i[j] = \perp$. Roughly, a correct process does not learn the value proposed by process G_j, if G_j is faulty[1].

We now give and prove the properties of our BC algorithm. We omit the proof that all processes that invoke decide agree on the vector $\mathbf{v}$ (this includes all correct processes, if $t < n/2$). Readers interested in this proof should refer to [147]. As discussed above, our algorithm inherently satisfies this property, given $t < n/2$. To summarize, our algorithm satisfies non-triviality, validity, and biased agreement properties of BC. Furthermore, it satisfies the early stopping property, i.e., every correct process decides in at most $\min(f+2, t+1)$ rounds.

[1] With different implementations of the decide procedure, the algorithm solves different problems. For example, if $\mathsf{decide}(\mathbf{v})$ would return a minimum of all (non-$\perp$) coordinates, the algorithm would solve consensus, which is precisely the case in [147].

The early stopping property is inherited from the original algorithm. Consider non-triviality. Assume that all processes are correct and all propose 1. In this case, all processes agree on vector $\mathbf{v} = 1^n$. Therefore decide returns 1 at every process. Consider now biased agreement and note that, if any process decides 1 it must have invoked decide and $\mathbf{v} = 1^n$. This implies that every correct process invokes decide, evaluates the same vector $\mathbf{v}$ that processes agreed on and, therefore, returns 1. Consider now validity. If some process G_j proposed $v_j = 0$, every process G_i that invokes decide (if any) has $\mathbf{v}_i[j] = 0$ or $\mathbf{v}_i[j] = \bot$, as processes agree on the vector $\mathbf{v}$ and the coordinate j of $\mathbf{v}$ is either $\bot$ or v_j. Therefore no process can decide 1. Note that validity holds for any t.

3.3.3 Gracefully Degrading Fair Exchange

The impossibility of solving FE (deterministically) if half of the processes can be dishonest, motivates the introduction of the following variant of the problem.

Definition 5 (gracefully degrading fair exchange). *An algorithm solves gracefully degrading fair exchange (GDFE) if it satisfies the following properties:*

- *The algorithm always satisfies the termination and completeness properties of fair exchange.*
- *If a majority of pirates are honest, then the algorithm also satisfies the fairness property of fair exchange.*
- *Otherwise (if there is no honest majority), the algorithm satisfies fairness with a probability p $(0 < p < 1)$ such that the probability of symmetric termination $(1 - p)$, i.e., the probability of unfairness, can be made arbitrarily low.*

3.4 A Gracefully Degrading Fair Exchange Protocol

3.4.1 Description

Our GDFE algorithm is described in Figure 3.3 (page 44). We assume that all processes involved in the algorithm know each other. The process with the lowest number is the initiator. We also assume a synchronous communication model [125] in the security subsystem. Basically, our algorithm can be viewed as an extension of the algorithm in Figure 3.1 (page 35), i.e., our reduction of deterministic fair exchange to biased consensus. However, whereas the algorithm in Figure 3.1 is made of an initialization phase followed by an agreement (BC) phase, the algorithm in Figure 3.3 introduces a fake phase between these two phases. This is the key to graceful degradation, i.e., minimizing the probability of unfairness in the case when $t \geq n/2$. Basically, we do not run the BC algorithm immediately after the exchange of items (i.e., unlike in Figure 3.1), but at some randomly picked round. In the meantime the processes exchange fake messages and, if necessary, react to the behavior of pirates. If any process detects a pirate misbehavior, e.g., a message omission, it aborts the algorithm immediately and does not participate in BC. It is important to notice that the underlying BC algorithm guarantees that no process decides 1 if some process does not participate in the algorithm (this missing process might have proposed 0). This is the way of penalizing any pirate that misbehaves in the first two phases of the algorithm.

The early stopping property of the underlying BC algorithm is essential for minimizing the probability for the adversary to violate fairness (as we discuss in the next subsection): in short, the early stopping BC algorithm we consider has two vulnerable rounds: if the adversary misses them, BC and the corresponding exchange terminate successfully. In addition, the introduction of dummy messages within the BC algorithm is necessary to solve the security requirement of our gracefully degrading fair exchange algorithm.

In our BC algorithm (Figure 3.2), every process in every round sends exactly one message to every other process, but some (dummy) messages are tagged to be disregarded by the receiving process. This is necessary in order to make sure that the adversary has no means distinguishing the fake phase from the agreement phase (i.e., the BC algorithm). We make use of the same communication pattern in both phases, i.e., the same distribution and sizes of the exchanged messages: every process sends a fixed-size message to every other process in every round, both in the fake phase and in the BC algorithm. Messages in fake phase are therefore padded to the size of BC message before sending. Hence, the adversary is not able to determine when BC starts, neither by observing when guardian angels send and receive messages, nor by observing the size of these messages.

3.4.2 Correctness of GDFE Algorithm

Theorem 7. *The algorithm shown in Figure 3.3 solves gracefully degrading fair exchange.*

Proof. The termination property is guaranteed by the fact that we consider a synchronous system and the termination property of BC. Consider completeness and assume that all participating pirates are honest and all items match their descriptions. All guardian angels will enter and exit the fake phase having $v = 1$, so all guardian angels will BCpropose 1. By the non-triviality property of BC every module returns 1 and subsequently returns the expected item to its pirate.

Now we consider fairness. It is important here to recall that the guardian angels are tamper-proof and no information leaks from them apart from what is explicitly released through their interface. We first prove a preliminary lemma. For convenience, if the guardian angel returns $\perp$ to its pirate, we say that the guardian angel aborts the GDFE algorithm.

Lemma 1. *If the first round in which some guardian angel G_j aborts the GDFE algorithm is round i $(0 \leq i < C)$, then at the end of round $i+1$ every guardian angel has aborted the GDFE algorithm.*

Proof. Because G_j has aborted the GDFE algorithm at the end of round i, no guardian angel will receive G_j's v in round $i+1$. Indeed, every guardian angel will abort the algorithm latest at the end of round $i+1$ (some modules might have aborted the algorithm in round i, like G_j). □

Consider the case in which the first misbehavior of some of the dishonest pirates occurs in the round i where $0 \leq i < C$ (misbehavior in round 0 includes the initiator's misbehavior or some dishonest pirate sending the wrong item). According to Lemma 1, by the end of the round $i+1 \leq C$, all guardian angels will abort the algorithm, so fairness is preserved.

Note that Lemma 1 does not hold for the C-th round. Some dishonest pirates can cut the channels for the first time in that round in such way that some guardian angels receive all messages and some do not. Hence some modules will BCpropose 1 and others will abort the algorithm at the end of round C and will not participate in BC. Because the modules that invoked consensus cannot distinguish this run from the run in which some faulty module proposed 0 and failed immediately in such way that it did not send or receive any message, all guardian angels that had invoked BC will return 0. At the end, none of the pirates gets the item.

The last possibility is that the first misbehavior occurs during the execution of the BC. This means that every guardian angel has proposed 1 to BC. If there is a majority of honest pirates, the biased agreement property of BC guarantees fairness. Indeed, fairness can be violated only if some guardian angel returns the expected item to its pirate, while some correct guardian angel returns $\perp$ to its honest pirate. It is obvious that this would be possible only if some guardian angel returns 1 from BC, while some correct guardian angel returns 0 which contradicts the biased agreement property.

If the adversary controls half or more of the pirates, fairness could be violated if and only if the adversary cuts one of the first two rounds of BC. However, this could occur only if the adversary successfully guesses in which round BC starts. Indeed, because our BC algorithm is early stopping, in order to succeed, the adversary must cut one of the first two rounds of BC and this has to be its first misbehavior in a particular algorithm run. In the following, we prove that if this case occurs, i.e., if there is no honest majority, probability of unfairness can be made arbitrarily low by choosing an appropriate distribution of the random number of fake rounds.

The number of rounds C in the second phase of the algorithm is chosen randomly by the initiator of the exchange according to a given distribution $(\beta_0, \beta_1, \dots)$ i.e., $\Pr(C = i) = \beta_i$. We assume this distribution to be public. The adversary performs the attack in a given round by dropping a certain subset of messages sent to, or received by, the pirates it controls, i.e., by cutting the channels. When the adversary cuts channels at more than $n/2$ pirates in the same round, we say that he *cuts the round.* Since the adversary does not know in which round BC starts, the best attack consists in choosing a value i according to the distribution $(\beta_0, \beta_1, \dots)$, starting from which adversary cuts all the rounds until the end of the exchange. Cutting messages at less that $n/2$ pirates, or cutting non-consecutive rounds, cannot improve the probability of success of the adversary.

We define the *probability of unfairness* $\Gamma_{(\beta_0,\beta_1,\dots)}$ as the maximum probability that an adversary succeeds, given the distribution $(\beta_0, \beta_1, \dots)$, and the *average complexity* in terms of number of fake rounds as $\Lambda_{(\beta_0,\beta_1,\dots)} = \sum_{i\geq 1} i\beta_i$.

Lemma 2. *Let $(\beta_0, \beta_1, \dots)$ denote the probability distribution of the value C. The probability of unfairness (for the algorithm in Figure 3.3) is*

$$\Gamma_{(\beta_0,\beta_1,\dots)} = \max_{i\geq 0}(\beta_i + \beta_{i+1}).$$

Proof. Let γ_i be the probability that the attack succeeds if it starts at round i $(i > 0)$. We already know that $\gamma_{i\leq C} = 0$ and that $\gamma_{i>C+2} = 0$. We have therefore:

$$\gamma_1 = \beta_0,\ \gamma_2 = \beta_0 + \beta_1,\ \gamma_3 = \beta_1 + \beta_2, \dots,\ \gamma_i = \beta_{i-2} + \beta_{i-1}, \dots$$

According to the probability distribution $(\beta_0, \beta_1, \dots)$, the maximum probability of unfairness $\Gamma_{(\beta_0,\beta_1,\dots)}$ is therefore

$$\Gamma_{(\beta_0,\beta_1,\dots)} = \max_{i>0}(\gamma_i) = \max_{i\geq 2}(\beta_0, \beta_{i-2} + \beta_{i-1}) = \max_{i\geq 0}(\beta_i + \beta_{i+1}).$$

□

The following example illustrates Lemma 2.

Example 3. *If* $(\beta_0, \beta_1, \dots, \beta_\kappa)$ *is the uniform distribution on the interval* $[0, \kappa]$, *then* $\beta_i = \frac{1}{\kappa+1}$ *if* $0 \leq i \leq \kappa$ *and* $\beta_i = 0$ *otherwise. We have therefore* $\Gamma_{uniform} = \max_{0\leq i\leq \kappa-1}(\beta_i+\beta_{i+1}) = \frac{2}{\kappa+1}$ *and the average complexity in terms of fake rounds is* $\Lambda_{uniform} = \frac{\kappa}{2}$ *in this case.*

Thus, we have proven that termination and completeness are ensured. Moreover, we have also proven that fairness is deterministically ensured when a majority of pirates are honest. If we do not have a majority of honest pirates, then the probability of unfairness can be made arbitrarily low according to Lemma 2, which concludes the proof of Theorem 7. □

3.4.3 Optimal Probability Distribution

We define the probability distribution that we call *bi-uniform*, as well as the *optimal* probability distribution for the algorithm in Figure 3.3.

Definition 6 (bi-uniform probability distribution). *We say that* $(\beta_0, \beta_1, \dots)$ *is a bi-uniform probability distribution of parameter* t *on the interval* $[0, \kappa]$ *if* $\forall i \geq 0,\ \beta_i + \beta_{i+1} = \frac{1}{\lceil\frac{\kappa+1}{2}\rceil}$ *and* $\beta_1 = t$ *if* κ *is odd, and* $\beta_1 = 0$ *if* κ *is even.*

Definition 7 (optimal probability distribution). *We say that a probability distribution* $(\beta_0, \beta_1, \dots)$ *is optimal (for the algorithm in Figure 3.3) if there is no other probability distribution* $(\beta'_0, \beta'_1, \dots)$ *such that* $\exists\Gamma > 0,\ \forall i \geq 0,\ \beta_i + \beta_{i+1} \leq \Gamma,\ \beta'_i + \beta'_{i+1} \leq \Gamma$ *and* $\Lambda_{(\beta'_0,\beta'_1,\dots)} < \Lambda_{(\beta_0,\beta_1,\dots)}$.

In other words, a probability distribution $(\beta_0, \beta_1, \dots)$ is optimal if there is no: (1) probability distribution $(\beta'_0, \beta'_1, \dots)$ and (2) probability of unfairness Γ such that the average complexity related to $(\beta'_0, \beta'_1, \dots)$, in terms of the number fake rounds, is lower that the average complexity related to $(\beta_0, \beta_1, \dots)$. Note that Definition 6 implies that, for a bi-uniform distribution,

$$\beta_i = \frac{1}{\lceil\frac{\kappa+1}{2}\rceil} - t \text{ if } i \text{ is even and } \beta_i = t \text{ if } i \text{ is odd.}$$

Moreover, t is necessarily equal to 0 if κ is even and $0 \leq t \leq \frac{1}{\lceil\frac{\kappa+1}{2}\rceil}$ if κ is odd. The following theorem states our optimality result in terms of probability of unfairness.

Theorem 8. *The optimal probability distribution (for the algorithm in Figure 3.3) is the bi-uniform probability distribution of parameter 0. Moreover, if the distribution is defined on* $[0, \kappa]$ *with* κ *even, the probability of unfairness is* $\Gamma_{bi\text{-}uniform} = \frac{2}{\kappa+2}$ *and the average complexity, in terms of the number of fake rounds, is* $\Lambda_{bi\text{-}uniform} = \frac{\kappa}{2}$.

Proof. First, we prove that if the probability distribution $(\beta_0, \beta_1, \dots)$ is optimal, then it is bi-uniform. In order to prove this assertion, we give Lemma 3.

Lemma 3. *Let $(\beta_0, \beta_1, \dots)$ denote the probability distribution of the value k. Let γ_i be the probability that the attack succeeds if it starts at round i $(i > 0)$. We have $\sum_{i\geq 1}\gamma_i = 2$.*

Proof. We have $\gamma_1 = \beta_0$, $\gamma_2 = \beta_0 + \beta_1$, $\gamma_i = \beta_{i-2} + \beta_{i-1}$, $\dots$ We have therefore

$$\sum_{i\geq 1}\gamma_i = \beta_0 + \sum_{i\geq 2}(\beta_{i-2}+\beta_{i-1}) = \left(\sum_{i\geq 0}\beta_i\right) + \left(\beta_0 + \sum_{i\geq 1}\beta_i\right) = 2\sum_{i\geq 0}\beta_i = 2.$$

□

Let Γ be the probability of unfairness and Λ by the average complexity in terms of number of fake rounds. We have $\Gamma = \max_{i\geq 0}(\beta_i + \beta_{i+1}) = \max_{i\geq 2}(\gamma_i)$ and $\Lambda = \sum_{i\geq 1} i\beta_i$. Since $\sum_{i\geq 1}\gamma_i$ is constant, $\sum_{i\geq 2} i\gamma_i$ is obviously minimum when the first γ_is are maximum, that is equal to Γ. Indeed, suppose there exists $j \geq 0$ such that

$$\gamma_i = \Gamma \text{ if } i < j \text{ and } \gamma_j < \Gamma, \tag{3.1}$$

then $\exists\epsilon > 0$ such that $\gamma_j = \Gamma - \epsilon$. Thus we have

$$\sum_{i\geq 2} i\gamma_i = \sum_{\substack{i\geq 2\\ i\neq j}} i\gamma_i + j(\Gamma-\epsilon) + \ell\epsilon$$

where $\ell > j$ (because $\ell \leq j$ contradicts Equation 3.1). So $j(\Gamma - \epsilon) + \ell\epsilon > j\Gamma$, implying that if γ_is are not maximum then $\sum_{i\geq 2} i\gamma_i$ is not minimum. Since $\gamma_0 = \beta_0$ and $\forall i \geq 2$ $\gamma_i = \beta_{i-2} + \beta_{i-1} = \Gamma$, we have

$$\begin{aligned}\sum_{i\geq 1} i\gamma_i &= \beta_0 + \sum_{i\geq 2} i\beta_{i-1} + \sum_{i\geq 2} i\beta_{i-2}\\ &= \beta_0 + \left(\sum_{i\geq 1} i\beta_i + \sum_{i\geq 0}\beta_i - \beta_0\right) + \left(\sum_{i\geq 1} i\beta_i + 2\sum_{i\geq 0}\beta_i\right) = 2\sum_{i\geq 1} i\beta_i + 3 = 2\Lambda + 3.\end{aligned}$$

So Λ being maximum implies that $\forall i \geq 0$ $\beta_i + \beta_{i+1} = \Gamma$ which further implies that $(\beta_0, \beta_1, \dots)$ is bi-uniform. Note that if $(\beta_0, \beta_1, \dots, \beta_\kappa)$ is a *finite* probability distribution, then $\gamma_i = \beta_{i-2} + \beta_{i-1}$ for $2 \leq i \leq \kappa + 1$, $\gamma_{\kappa+2} = \beta_\kappa$ and $\gamma_i = 0$ if $i > \kappa + 2$.

We now prove that the bi-uniform probability distribution of parameter t is optimal when $t = 0$. As previously, we argue that $\sum_{i\geq 1} i\beta_i$ is minimum when the first β_is are maximum. Since $\forall i \geq 0$ $\beta_i + \beta_{i+1} = \Gamma$, the probability distribution is optimal if $\beta_i = 0$ when i is odd, that is when $t = 0$. Since $\beta_i = 0$ when i is odd, we suppose that κ is even. So, if $(\beta_0, \beta_1, \dots, \beta_\kappa)$ is a bi-uniform probability distribution of parameter 0 such that κ is even, we have

$$\Gamma_{\text{bi-uniform}} = \frac{1}{\lceil \frac{\kappa+1}{2} \rceil} = \frac{2}{\kappa+2}$$

$$\text{and} \quad \Lambda_{\text{bi-uniform}} = \sum_{i\geq 1} i\beta_i = \sum_{\substack{i\geq 1\\ i \text{even}}} i\Gamma_{\text{bi-uniform}} = \frac{(\kappa+2)\kappa}{4}\Gamma_{\text{bi-uniform}} = \frac{\kappa}{2}.$$

□

```
BCpropose($v_i$)
  $\mathbf{v}_i := \bot^n$; $\mathbf{v}_i[i] := v_i$; $new_i := \bot^n$; $\mathbf{new}_i[i] := v_i$;
  $locked_i := \emptyset$; $suspected_i := \emptyset$;
  for $r := 1, 2, ..., t+1$ do
  begin_round
    foreach $p_j$ do
      if $p_j \in suspected_i$ then $dummy := 1$ else $dummy := 0$ endif
       send $(new_i, locked_i, dummy)$ to $G_j$
    enddo
    $\mathbf{new}_i := \bot^n$
    foreach $G_j \notin suspected_i$ do
      if $(\mathbf{new}_j, locked_j, dummy = 0)$ has been received from $G_j$ then
        foreach $m \in [1 \ldots n]$ do
          if $(\mathbf{new}_j[m] \neq \bot)$ and $(\mathbf{v}_i[m] = \bot)$ then
            $\mathbf{v}_i[m] := \mathbf{new}_j[m]$; $\mathbf{new}_i[m] := \mathbf{new}_j[m]$
            $locked_i := locked_i \cup locked_j$
          endif
        enddo
      else
        if $(G_j \notin locked_i)$ then $suspected_i := suspected_i \cup \{G_j\}$ endif
      endif
    enddo
    if $(|suspected_i| > t)$ then return 0 endif
    if $(G_i \notin locked_i)$ then
      if $(r > |suspected_i|)$ or $(locked_i \neq \emptyset)$ then $locked_i := locked_i \cup \{G_i\}$ endif
    else
      if $(|locked_i| > t)$ then decide($\mathbf{v}_i$) endif
    endif
  end_round   decide($\mathbf{v}_i$)

decide($\mathbf{v}$)
  if $(\exists m, 1 \leq m \leq n$, s.t. $(\mathbf{v}[m] = \bot)$ or $(\mathbf{v}[m] = 0))$ then
    return 0
  else
    return 1
  end
```

Figure 3.2: Early stopping biased consensus protocol: code of process G_i

```
GDFairExchange($myitem$, $description$, $source$, $destination$)
  if $\langle G_i$ is $initiator \rangle$ then % $initialization$ phase (round 0)
    $\langle$pick a random number $C$ according to a given distribution$\rangle$
    foreach $G_j \neq destination$ do send $(\perp, C)$ to $G_j$ enddo
    send $(myitem, C)$ to $destination$
  else
    send $(myitem, \perp)$ to $destination$
  endif
  if $((item, *)$ has been received from $source)$ and $(item$ matches $description)$
    and $((*, C)$ has been received from $initiator)$ then
    $v_i := 1$; $C_i := C$; $item_i := item$
  else
    return $\perp$
  endif
  for $round := 1, 2, \ldots, C_i$ do % fake phase ($C$ rounds)
    send $\langle$padded $v_i\rangle$ to all
    if not($(v)$ has been received from all processes) then return $\perp$ endif
  enddo
  $v_i :=$ BCpropose$(v_i)$ % $agreement$ phase – biased consensus
  if $(v_i = 1)$ then return $item_i$ else return $\perp$ endif
```

Figure 3.3: Gracefully degrading fair exchange protocol: code of process G_i

Two-Party Fair Exchange in Pretty Honest Neighborhood

CHAPTER FOUR

Up until now, the two-party protocols considered the main participants were isolated in their own world. They were able to communicate only with the other main participant and possibly with a TTP. The basic idea presented in this chapter [30] relies on the observation that two entities willing to exchange information are usually not isolated in a closed environment but, instead, are part of a network involving other peer entities. Thus, the fairness of the exchange may rely on the honesty of their "neighbors": when the exchange runs well, no communication overhead is required to any third party, but when a conflict occurs, neighbors are requested to restore fairness, by recovering the unreceived item. Thus we have a kind of distributed optimistic protocol.

For that, we use a cryptographic building block introduced by Stadler [162] in 1996 called *publicly verifiable secret sharing.* Secret sharing, suggested independently by Blakley [41] and Shamir [160] in 1979, allows to share a secret among participants such that only certain subsets of participants can recover the secret. Publicly verifiable secret sharing brings an additional property: anybody is able to check whether or not the distributed shares are correct.

In what follows, we define the protocol requirements, and the communication and threat models. In Section 4.2, we briefly recall the publicly verifiable secret sharing concept and describe a practical protocol. We then propose an optimistic two-party fair exchange protocol based on a generic publicly verifiable secret sharing. A security analysis is finally provided in Section 4.4.

4.1 Security Model

4.1.1 Requirements

Definition 1 provided in Chapter 1 encloses the properties of completeness, fairness, and termination. Two-party fair exchange in a pretty honest neighborhood requires additional properties, called *privacy* and *optimism*. We therefore complete Definition 1 as follows.

Definition 8 (exchange protocol). *An exchange protocol between two parties A and B is a protocol in which A and B own some items m_A and m_B respectively and aim at exchanging them. We say that the protocol ensures:*

- completeness *if A gets m_B and B gets m_A at the end of the protocol when there is no malicious misbehavior;*
- fairness *if the protocol terminates so that either A gets m_B and B gets m_A, or A gets no information about m_B and B gets no information about m_A;*
- termination *if A and B eventually end;*
- privacy *if no other participant gets any information about m_A and m_B;*
- optimism *if no other participant is involved when A and B are honest.*

The protocol described in this chapter fulfills Definition 8.

4.1.2 Communication and Threat Model

Definition 9 (environment). *We say that two entities A and B are in the same environment if and only if A and B are able to communicate through a channel which guarantees confidentiality, authentication, integrity, timeliness, and sequentiality. We let Env_A denote the set of all entities that are in the same environment as A.*

Remark 1. *The relation $B \in Env_A$ between A and B is symmetric but not transitive due to the timeliness requirement.*

As usual, we will not consider the case where both A and B are dishonest since they obviously always have the ability to halt the exchange on an unfair termination in this case. External participants can be classified in the following sets:

- $\mathcal{P}_1$: participants who honestly collaborate with both A and B.
- $\mathcal{P}_2$: participants who may harm A by colluding with B.
- $\mathcal{P}_3$: participants who may harm B by colluding with A.
- $\mathcal{P}_4$: participants who do not collaborate at all.

Note that $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and $\mathcal{P}_4$ form a partition of the external participants. We denote $b_i = |\mathcal{P}_i|$ where $1 \leq i \leq 4$. We assume that participants cannot move from one set to another. Therefore, we focus on the "honesty status" at the time of the recovery protocol only.

We highlight that our model deals with only one virtual adversary that manages all the misbehaving external participants. This is the common way to model the adversary in multi-party protocols. This is also the strongest adversary model. Thus, we never consider several *independent* adversaries among the external participants. This approach nevertheless would deserve to be explored.

4.1.3 Hypothesis

In what follows we assume that $B \in Env_A$; B knows a subset $\mathcal{P}$ of external participants in $Env_A \cap Env_B$ and a constant $T_{\max} < +\infty$ such that messages from B to any participant in $\mathcal{P}$ are always delivered within a time delay less than $T_{\max}$; $b_1 > 0$; and A and B know some constant k such that $b_2 < k \leq b_2 + b_1$. We give here two examples in order to illustrate this assumption.

Example 4. *If A and B know that there is a majority of honest participants in the network, i.e., $b_1 > \frac{n}{2}$ then we take $k = \lceil \frac{n}{2} \rceil$.*

Example 5. *If A knows that at least 40% of the network is honest with him (i.e., $b_1 + b_3 \geq \frac{2n}{5}$) and B knows that at least 70% of the network is honest with him (i.e., $b_1 + b_2 \geq \frac{7n}{10}$) then we can take k such that $\lfloor \frac{6n}{10} \rfloor < k \leq \lceil \frac{7n}{10} \rceil$. For instance, if $n = 100$, k is chosen such that $60 < k \leq 70$. We show in Section 4.3 that k is actually the threshold of the secret sharing.*

4.2 Publicly Verifiable Secret Sharing

4.2.1 Basic Secret Sharing and Variants

Secret sharing [41, 160] allows sharing a secret m among several participants such that only some specific subsets of participants can recover m by collusion. In the Shamir's secret sharing scheme, there is a threshold k so that only subsets of at least k participants can reconstruct m. A drawback of the Shamir 's scheme is that participants cannot verify that the distributed shares effectively allow to recover the secret m. In other words, the basic secret sharing scheme assumes that the dealer is not malicious. *Verifiable* secret sharing [55, 83, 158, 162] resists a malicious dealer who sends wrong shares: each participant can indeed check its own share. In *Publicly* verifiable secret sharing [158, 162], anybody can perform this verification, not only the participants. Below we describe a model for non-interactive publicly verifiable secret sharing.

Distribution Stage

The dealer generates the shares m_i of m and then publishes the encrypted values $E_i(m_i)$ such that only the participant P_i is able to decrypt $E_i(m_i)$. The dealer also publishes an information Δ containing $\theta = \mathcal{W}(m)$ where $\mathcal{W}$ is a one-way function. This information allows proving that the distributed shares are correct, i.e., they allow recovering some m such that $\mathcal{W}(m) = \theta$.

Verification Stage

Given the P_is' public keys, the $E_i(m_i)$s, Δ, and a verification algorithm, anybody can verify that the shares allow recovering some m such that $\mathcal{W}(m) = \theta$.

Reconstruction Stage

The participants decrypt their share m_i from $E_i(m_i)$ and pool them in order to recover m.

4.2.2 A Practical Publicly Verifiable Secret Sharing Scheme

We describe in this section a practical publicly verifiable secret sharing scheme which has been proposed by Stadler [162] in 1996. The main idea consists in sharing items among n participants such that k participants are enough to recover these items in case of conflict. This scheme relies on both Elgamal's public key cryptosystem [68] and on the double discrete logarithm assumption [162]. Let p be a large prime number so that $q = (p-1)/2$ is also prime, and let $h \in (\mathbf{Z}/p\mathbf{Z})^*$ be an element of order q. Let G be a group of order p, and let g be a generator of G such that computing discrete logarithms to the base g is difficult. Let $m \in \mathbf{Z}/p\mathbf{Z}$ be the secret and let $\mathcal{W}(m) = g^m$. As in Shamir's scheme, we assume that a publicly known element $x_i \in \mathbf{Z}/p\mathbf{Z}$, $x_i \neq 0$, is assigned to each participant P_i. We assume also that each participant P_i owns a secret key $z_i \in \mathbf{Z}/q\mathbf{Z}$ and the corresponding public key $y_i = h^{z_i} \bmod p$.

Distribution Stage

The dealer chooses random elements $a_j \in \mathbf{Z}/p\mathbf{Z}$ $(j = 1, ..., k-1)$ and publishes the values $\xi_j = g^{a_j}$ $(j = 1, ..., k-1)$ in Δ. Then he securely computes the share

$$m_i = m + \sum_{j=1}^{k-1} a_j x_i^j \bmod p \tag{4.1}$$

for P_i and publishes the value g^{m_i} in Δ $(1 \leq i \leq n)$. He uses the Elgamal encryption: he chooses a random value $\alpha_i \in \mathbf{Z}/q\mathbf{Z}$, computes the pair

$$E_i(m_i) = (\sigma_i^1, \sigma_i^2) = (h^{\alpha_i}, m_i^{-1}.\, y_i^{\alpha_i}) \bmod p$$

and publishes it in Δ $(1 \leq i \leq n)$. The precise content of Δ is described below.

Verification Stage

The first step of this procedure consists in verifying the consistency of the shares. Anybody is able to perform this step by checking whether

$$g^{m_i} = g^m. \prod_{j=1}^{k-1} \xi_j^{x_i^j},$$

obtained by exponentiating Equation 4.1, is satisfied in G (note that Δ includes g^m, g^{m_i}, ξ_j). The second step consists in verifying that the pairs (σ_i^1, σ_i^2) really encrypt the discrete logarithms of public elements g^{m_i}. This verification is based on the fact that the discrete logarithm of $\sigma_i^1 = h^{\alpha_i}$ to the base h equals the double discrete logarithm of $g^{m_i\sigma_i^2} = g^{(y_i^{\alpha_i})}$ to the bases g and y_i. One may give a zero-knowledge interactive verification procedure between the dealer and participants as described in Figure 4.1. We describe here the non-interactive version which is obtained by simulating the verifier by a hash function. We assume that the dealer randomly picked some values $w_{i,\ell} \in \mathbf{Z}/q\mathbf{Z}$ ($1 \leq \ell \leq L$ where $L \approx 100$ from [162]) for each share m_i and computed:

$$\begin{aligned} \delta_{i,\ell} &:= h^{w_{i,\ell}} \bmod p \\ \gamma_{i,\ell} &:= g^{y_i^{w_{i,\ell}}} \\ r_{i,\ell} &:= w_{i,\ell} - c_{i,\ell}\alpha_i \bmod q \end{aligned}$$

where $c_{i,\ell}$ denotes the ℓ-th bit of $c_i = \mathcal{H}(g^{m_i}\|\sigma_i^1\|\sigma_i^2\|\delta_{i,1}\|\gamma_{i,1}\|...\|\delta_{i,L}\|\gamma_{i,L})$ with $\mathcal{H}$ a hash function from $\{0,1\}^*$ to $\{0,1\}^L$. Participants have therefore to check for all ℓ whether

$$\delta_{i,\ell} = h^{r_{i,\ell}}{\sigma_i^1}^{c_{i,\ell}} \bmod p \text{ and } \gamma_{i,\ell} = (g^{1-c_{i,\ell}}g^{m_i c_{i,\ell}\sigma_i^2})^{y_i^{r_{i,\ell}}}.$$

Δ finally contains g^m, the g^{m_i}s, the $r_{i,\ell}$s, the $\delta_{i,\ell}$s, the $\gamma_{i,\ell}$s, and ξ_j.

Reconstruction Stage

Each participant P_i decrypts his own share m_i by computing

$$m_i = \frac{(\sigma_i^1)^{z_i}}{\sigma_i^2} \bmod p.$$

A subset of k participants can then recover m by using the Lagrange's interpolation formula.

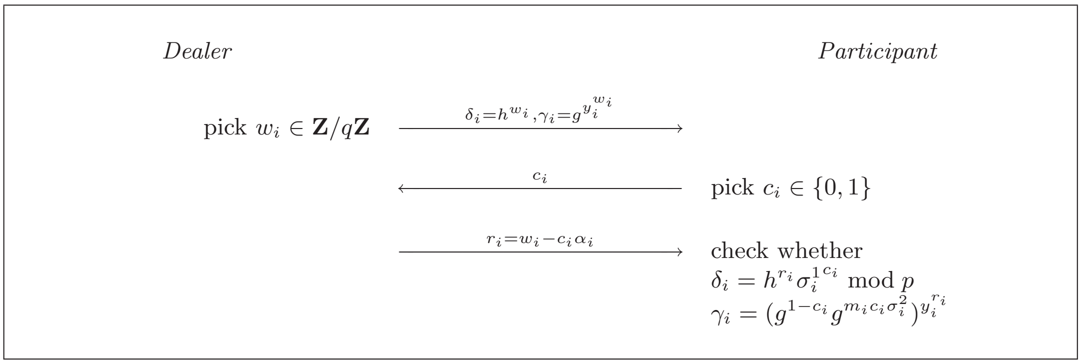

Dealer
Participant
pick $w_i \in \mathbf{Z}/q\mathbf{Z}$
$\delta_i = h^{w_i}, \gamma_i = g^{y_i^{w_i}}$
c_i
pick $c_i \in \{0,1\}$
$r_i = w_i - c_i\alpha_i$
check whether
$\delta_i = h^{r_i}{\sigma_i^1}^{c_i} \bmod p$
$\gamma_i = (g^{1-c_i}g^{m_i c_i \sigma_i^2})^{y_i^{r_i}}$

Figure 4.1: Interactive verification procedure

4.3 Description of the Optimistic Protocol

4.3.1 Primitives

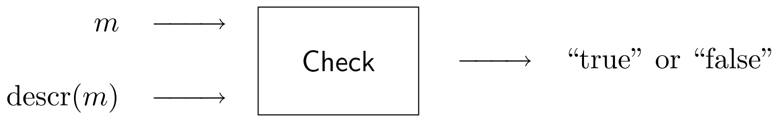
We define in this section some primitives that will be used in our protocol. These primitives are independent of the publicly verifiable secret sharing that is used.

Signature

We consider A's authentication function S_A which, given a message m, outputs the signed version $m' = S_A(m)$ and the corresponding verification function V_A. Note that S_A is either a signature with message recovery or the concatenation of the message with the signature.

Item Description

As it is typically assumed in the literature, we suppose that A and B have committed to their items beforehand. We therefore consider that A and B established a legal agreement linking the authentic human-readable descriptions of the items with their mathematical descriptions $\mathrm{descr}(m_A)$ and $\mathrm{descr}(m_B)$. For instance, $\mathrm{descr}(m) = \mathcal{W}(m) = g^m$. According to the fact that the authentic descriptions match the mathematical descriptions (a conflict at this layer can only be resolved by legal means), participants will be satisfied if they receive an item m which is consistent with its description $\mathrm{descr}(m)$. To check that, we consider the public contract $\Omega = S_A(A\|B\|\mathrm{descr}(m_A)\|\mathrm{descr}(m_B)\|D)$, where D is the expiration date after when the exchange has to be considered as null and void.

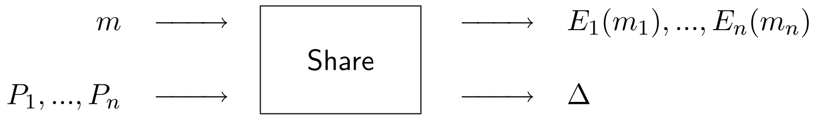


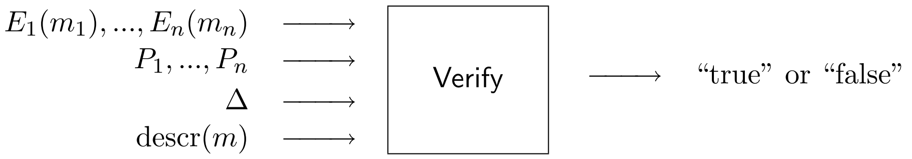
Encryption

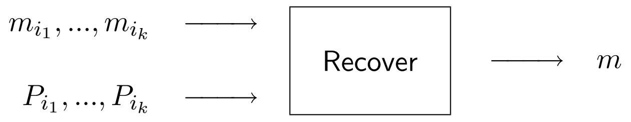
We consider P_i's encryption function E_i which, given a message m, outputs the encrypted message $m' = E_i(m)$ for P_i and the corresponding decryption function D_i that, given an encrypted message m', outputs the plain message $m = D_i(m')$.

Verifiable Secret Sharing

We consider a publicly verifiable secret sharing scheme using the functions Share, Verify, and Recover. Given a message m and some participants P_i $(1 \leq i \leq n)$, Share outputs the encrypted shares $E_1(m_1)$,..., $E_n(m_n)$, and the proof Δ, as described in Section 4.2.2; given a list of encrypted shares, a list of participants, Δ, and $\mathrm{descr}(m)$, Verify outputs "true" if the shares allow any subset of k participants to recover m and "false" otherwise; given some shares m_{i_1},..., m_{i_k} and participants P_i $(i \in \{i_1, ..., i_k\})$, Recover outputs the message m.

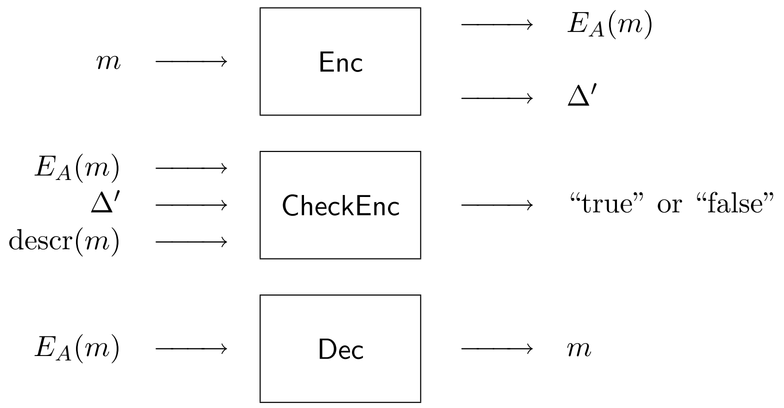






Verifiable Encryption

We describe the CheckEnc function which is pretty similar to the Check function except that its input is $E_A(m)$ rather than m. This function is used by the *external* participants. We will not detail this function for simplicity sake, but it could come from a publicly verifiable secret sharing with a single participant. Below, we only sketch the primitives related to CheckEnc:



4.3.2 Description of the Protocols

Two participants A and B wish to exchange items m_A and m_B in a set of participants $\mathcal{P}$ such that $\mathcal{P} \subset Env_A \cap Env_B$ with $|\mathcal{P}| = n$. Note that for simplicity sake, we exclude A and B from $\mathcal{P}$, although it is not mandatory; so $\mathcal{P}$ contains only external participants. Here are the exchange and recovery protocols.

Exchange Protocol

The exchange protocol, depicted in Figure 4.2, implies only the main participants, A and B, and consists in exchanging items m_A and m_B after a commitment step. This commitment gives B the ability to restore fairness of the exchange, helped by the external participants, in case of conflict with A.

- **Step 1:** A picks a random element a and computes b such that $m_A = a + b$. He computes Share$(a, P_1, ..., P_n)$ and sends $E_1(a_1), ..., E_n(a_n), \Delta, \Omega, b$ to B.
- **Step 2:** B checks that Verify$(E_1(a_1), ..., E_n(a_n), P_1, ..., P_n, \Delta, \text{descr}(a))$ is "true" where descr(a) is deduced from descr(m_A) (extracted from Ω) and b, e.g., $g^a = g^m \times g^{-b}$. If the test succeeds then he sends m_B to A; otherwise he just has to wait until the expiration date D to give up the exchange.
- **Step 3:** A checks that m_B is correctly running Check$(m_B, \text{descr}(m_B))$. If this is the case then A sends m_A to B. Otherwise, he has just to wait until the expiration date D in order to give up the exchange.

- **Step 4:** If B does not receive m_A or if $\mathsf{Check}(m_A, \mathrm{descr}(m_A))$ is "false" then he runs the recovery protocol.

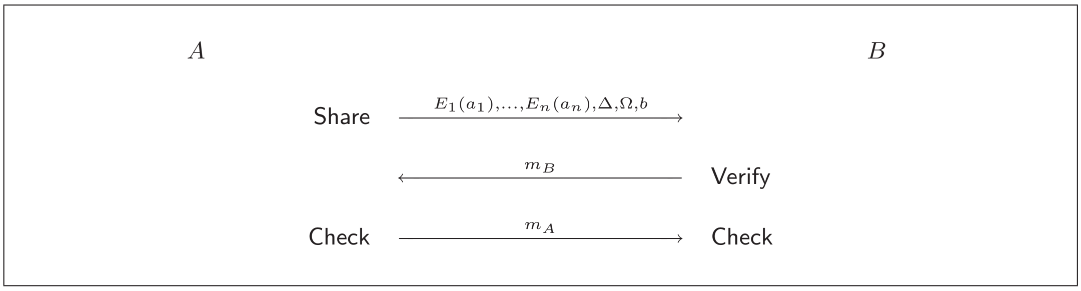


Figure 4.2: exchange protocol

Recovery Protocol

The recovery protocol depicted in Figure 4.3, is started before $D - T_{\max}$ by the recipient, B, when he is injured, that is if the third message of the exchange, m_A, is wrong or missing.

- **Step 1:** B encrypts m_B for A and sends $E_i(a_i)$, $E_A(m_B)$, Δ', and Ω to P_i.
- **Step 2:** P_i computes $\mathsf{CheckEnc}(E_A(m_B), \mathrm{descr}(m_B), \Delta')$ where $\mathrm{descr}(m_B)$ is extracted from Ω; if the output is "true" and if the expiration date, contained in Ω, has not expired, P_i sends a_i to B and $E_A(m_B)$ to A.
- **Step 3:** After having received k shares, B runs Recover. From a he computes $m_A = a + b$.

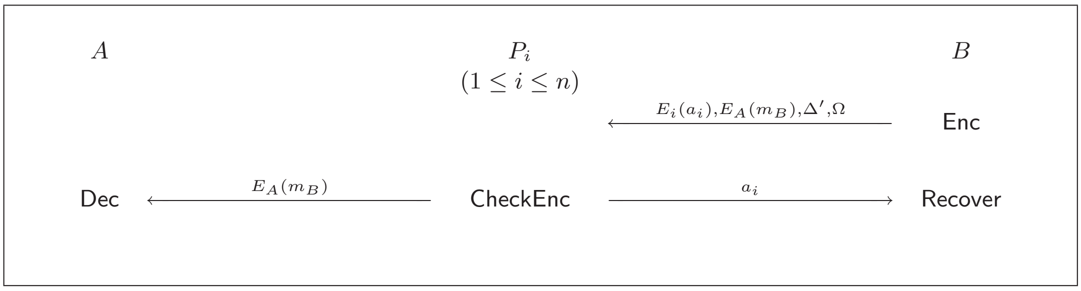


Figure 4.3: recovery protocol

Remark 2. *In existing optimistic fair exchange protocols, the TTP is stateful: the TTP keeps in memory whether the* recovery *or* abort *protocol has already been performed. Due to the present distributed architecture, this model cannot be used here and using expiration dates is preferred.*

4.4 Security Analysis

We prove in this section that the properties of completeness, fairness, termination, and privacy are respected, even in case of misbehaviors. We recall that the security parameter k, which is the threshold of the publicly verifiable secret sharing, is such that $b_2 < k \leq b_2 + b_1$. We defined in Section 4.1.2 the set of external participants $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and $\mathcal{P}_4$. We rewrite these definitions here according to our protocol defined in Section 4.3.2

- $\mathcal{P}_1$: participants who honestly collaborate with both A and B;
- $\mathcal{P}_2$: participants P_i such that when B sends $E_i(a_i)$ to $P_i \in \mathcal{P}_2$, P_i decrypts $E_i(a_i)$ and sends a_i to B (even if the date is expired) but does not send m_B to A;
- $\mathcal{P}_3$: participants P_i such that when B sends $E_i(a_i)$ to $P_i \in \mathcal{P}_3$, P_i sends $E_A(m_B)$ to A (even if the date is expired) but does not decrypt $E_i(a_i)$ for B;
- $\mathcal{P}_4$: participants who do not collaborate at all. We denote M_1, M_2, and M_3 the three messages of the exchange protocol consisting respectively of $(E_1(a_1), ..., E_n(a_n), \Delta, \Omega, b)$, m_B, and m_A.

4.4.1 Completeness

Proving that the protocol is complete when A and B are honest is straightforward. In case of late discovery of M_3 due to communication protocol reasons, B runs the recovery protocol. Since $b_2 < k \leq b_2 + b_1$ we have at least k participants who will collaborate with B and at least one who will collaborate with A.

4.4.2 Fairness

We saw in the previous section that the protocol is fair if both main participants are honest. Furthermore, we explained in Section 4.1.2 that the case where both main participants are dishonest is not relevant. We consider therefore the two following cases only.

A is Honest and B is Dishonest

Since A is honest, M_1 is correctly formed. On the other hand M_2 is wrong (or missing) otherwise both A and B would be honest. Here A can detect that M_2 is wrong using Check(m_B,descr(m_B)); therefore he does not transmit M_3 and waits for D. If B does not run the recovery protocol then nobody can obtain anything valuable on the expected items and the exchange is trivially fair. If B starts the recovery protocol after D then he cannot obtain m_A since $b_2 < k$. If B starts the recovery protocol before D (note that if he only contacts participants in $\mathcal{P}_2$ or $\mathcal{P}_4$ before D, then we fall into the previous case), then A receives m_B from an external participant either in $\mathcal{P}_1$ or $\mathcal{P}_3$; therefore the protocol is fair if and only if B can obtain m_A, that is, if and only if $b_1 + b_2 \geq k$.

A is Dishonest and B is Honest

Since A is dishonest, we consider the case where M_1 is wrong (or missing) and the case where M_1 is correct but M_3 is not (or missing).

Firstly, if M_1 is wrong (or missing), the exchange will die after D. Indeed B can perform $\mathsf{Verify}(E_1(a_1), ..., E_n(a_n), P_1, ..., P_n, \Delta, \mathrm{descr}(a))$ and detect that M_1 is wrong; he therefore decides not to disclose m_B. Thus the exchange ends on a trivially fair termination after D.

Secondly, if M_1 is correct but M_3 is not (or missing): B can detect such a wrong M_3 using $\mathsf{Check}(m_A$,$\mathrm{descr}(m_A))$ and therefore start the recovery protocol. The fairness of the exchange thus relies on the external participant's ability to supply a to B, that is, if and only if $b_1 + b_2 \geq k$. The fairness is guaranteed since A has already received m_B in M_2.

4.4.3 Termination

Termination of the protocol is straightforward due to the expiration date D.

4.4.4 Privacy of the Protocol

If the recovery protocol is not carried out, then only information between A and B are exchanged and external participants receive nothing. If the recovery protocol is used, then some participants receive shares of a_i. However, although k participants colluding can recover a, they cannot recover m_A since they do not know b. Obviously, they cannot discover m_B either. Privacy here is greatly improved compared to previous optimistic fair exchange protocols where we usually assumed that the trusted third party is able to regenerate expected items.

4.4.5 Complexity of the Protocol

When both A and B are honest, the complexity in terms of exchanged messages is very small since only the three messages of the exchange protocol are sent. When somebody misbehaves, the complexity obviously increases since the recovery procedure is performed. In the worst case, the n external participants are contacted by B, each receives one message and sends at most two messages, so the complexity is only $O(3n)$ in terms of exchanged messages.

Part II

Radio Frequency Identification

Introduction to Radio Frequency Identification

CHAPTER FIVE

Often presented as the new technological revolution, radio frequency identification (RFID) allows us to identify objects or subjects with neither physical nor visual contact. We merely need to place a transponder (see page 61) on or in the object and query it remotely using a reader. Even though it has only been a few months since it has started to be covered by the media, this technology is fundamentally not new. It was used during the Second World War by the Royal Air Force to distinguish allied aircrafts from enemy aircrafts. It has also been used for many years in applications as diverse as: motorway tolls, ski lifts, identification of livestock and pets, automobile ignition keys, etc. Thus, there exists a whole range of RFID technologies that have very different purposes and characteristics.

Establishing a precise classification of the current RFID technologies is hard because their characteristics are numerous and non-independent. Thus, the answer to the question "What is an RFID device?" is not obvious. For example, one may think that an electronic anti-theft device is an RFID device. Conversely, one may think that a mobile phone or a Bluetooth headset is an RFID device. In our opinion, these devices cannot be considered as RFID devices because their primal purpose is not the identification of their bearer.

The boom that RFID enjoys today rests on the ability to develop very small and cheap transponders called "electronic tags". These tags only offer weak computation and storage capacities. They are passive, that is to say, they do not have their own battery, but take their power from the reader's electromagnetic field, which in turn implies that their communication distance is relatively short, i.e., a few meters in the best case. When they are outside the reader's field, the electronic tags are inert, incapable of carrying out any sort of calculation or communication. However, their low cost, expecting to reach 5 Euro cents each in the near future, and their small size, sometimes less than a square millimeter, gives them undeniable advantages that could be exploited in innumerable domains. Since they are so discreet, bearers may not even be aware they are holding any. This opens new security challenges. The present dissertation focuses on this kind of RFID technology.

In this chapter, we give an overview of the (powerless) RFID technology, thus supplying knowledge that will be required in the following chapters, when the security and cryptographic aspects will be tackled. First of all, we give a few examples of daily life applications where RFID is involved. In Section 5.2 and Section 5.3, we describe RFID architectures and the tags characteristics. Finally, in Section 5.4, we introduce the notion of identification.

5.1 Daily Life Examples

5.1.1 Supply Chains

Radio frequency identification represents a major step forward in relation to optical identification found on barcodes. In addition to the tags miniaturization that allows them to be placed right in the heart of objects, they can be scanned in large numbers, without having to place the object in the reader's optical beam. Let us note that each one possesses a unique identifier: whereas a barcode represents a group of objects, an electronic tag represents a single object. Management of stock and inventories in shops and warehouses is a prime domain for low cost tags. The American mass marketing giant Wal-Mart has recently begun requiring its main suppliers to put electronic tags in the pallets and packing cases that they deliver to it. The American Department of Defense has gone down the same route. Migros, leading figurehead in the Swiss market, would like to put an electronic tag in every product, thus making it possible to carry out real-time inventories using readers placed along the shelves and showing customers the contents of their trolley electronically. This would also allow a reduction in queues at the checkout, by replacing the charming cashier with an automatic device. However, shoppers will not be able to enjoy the fruits of this experiment for several years.

5.1.2 Tags in Libraries

The advantages of electronic tags can also be seen in libraries. Inserting a tag in each volume makes borrowing and returning books easier, because the user can carry out these operations himself. Better still, they can be completely automated by using readers on the exit gates. Besides the advantages that the library users can get from the system, the library staff see their job made easier; inventories can be carried out without taking books from the shelves, by automatically detecting missing or misfiled books, or even by using an automatic sorter for the returned volumes. Libraries such as K.U. Leuven (Belgium), Santa Clara (United States), Heiloo (The Netherlands), or Richmond Hill (Canada) already use this technology. According to the Swiss company "Bibliotheca RFID Library Systems AG" [39], specialist in this field, more than 100 million documents have been tagged throughout the world.

5.1.3 Subdermal Tags

Domestic animals identification by RFID is already a part of our daily lives. But subdermal tags are not only for animal use. A famous example is the nightclub "Baja Beach Club" in Barcelona, which suggests to its regulars that they have an electronic tag implanted as a means of identification. The tag is linked to a personal electronic dossier which allows the customer to settle their accounts and gain access to VIP areas. In Mexico, members of the Attorney General's Office have also been "tagged" with subdermal implants, which give them access to sensitive areas and information. The number of people who have been tagged remains unknown. The international press first announced 160 people, but then published a retraction, saying that only 18 people were involved in the electronic marking.

Injecting subdermal tags in prisoners is maybe not such a distant future plan. In California, Senate Bill 682 restricts the use of RFID-based ID documents except in certain cases, which concern for example people who are incarcerated in the state prison or people pursuant to court-ordered electronic monitoring.

5.1.4 Access Control

Nowadays, access control applications are often based on wireless tokens, e.g, cards or badges. The wireless property renders control easier: people do not have to take out their token, controls are systematic, damages to readers and in particular vandalism are reduced. Systematic controls force two people accessing the restricted area at the same time, both to pass the check. Beyond the primary goal of access control, the interest of systematic controls is to determine, in real time, the people present within a given area. It can be for safety reasons, e.g., when a building must be evacuated.

A particular example of access control can be found in the automobiles domain. Early applications were keyless entry: the driver opens and closes the car using a key fob that contains an active RFID tag. Passive entry systems appeared recently, e.g., on Renault Laguna, Mercedes-Benz S-class, CL-class, and Toyota Lexus LS430. The driver, who is carrying a passive RFID tag, simply approaches the car and the door automatically unlocks itself. Still further, today, many car keys have an RFID device integrated into them which activates the fuel injection system, thus preventing the vehicle from starting if the electronic tag is not present. The tag is activated when the key is inserted in the starting device and receives a challenge from the reader. Sending a correct response to this challenge is the *sine qua non* condition to starting the vehicle. In some cases, the (physical) ignition key is no longer required. The driver has just to press a button "start" to ignite the car: the driver has an RFID tag embedded into a card that can stay in his pocket.

5.2 RFID Architecture

As depicted in Figure 5.1, an RFID system consists of tags, readers and a back-end infrastructure. A tag is a transponder, that is, an electronic microcircuit equipped with an antenna, capable of communicating using radio waves. *A priori*, the reader is only a physical device whose functions are to supply energy to the tag, to send a request, to collect its response and send the latter to the back-end infrastructure. The back-end infrastructure has a database (denoted DB hereafter) that contains tag identifiers and information concerning these tags.

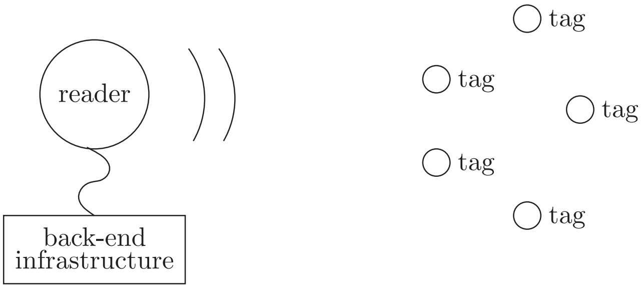


Figure 5.1: Radio frequency identification architecture

5.2.1 Reader and Back-End Infrastructure

From the RFID systems security analysis point of view, either the reader is a simple physical device and is not of significant importance, or the reader has sensible data and we would have to secure both the reader and the communication between the reader and the back-end infrastructure. For this reason, the readers and back-end infrastructure are usually considered as a single entity, called *reader* or *system* by convention. Hereafter, we will use either reader or system when referring to the group that comprises the back-end infrastructure and its corresponding readers.

Let us note that it could be advantageous to separate these two distinguished entities. This would allow conceiving and analyzing protocols where the back-end infrastructure forwards calculations or data to the physical readers. Ensuring that the back-end infrastructure does not become the system's bottleneck is an important problem that will have to be dealt with in the future.

5.2.2 Legitimate Tags and Readers

We now define the “legitimate” concept. Given a system or a reader, we say that a tag is *legitimate* if it is registered in the system's database. Given a tag, we say that a system or a reader is legitimate if the tag is registered in its database. A tag or a reader that is not legitimate is called *foreign*.

5.2.3 System Topology

Intuitively, an RFID system should handle several tags (e.g., physical access control in a company), possibly even several million tags (e.g., tagged books in library). However, there also exist many deployed systems that only have a single tag, as in the car ignition keys. This leads to the reader-to-tag topologies so-called *one-to-many* and *one-to-one*. The considered topology can have a big influence on the identification performances and thus on the cryptographic tools to be implemented. In the following, we shall see, and more specifically in Chapter 9, that certain protocols are not scalable and thus cannot be used in systems based on a one-to-many topology.

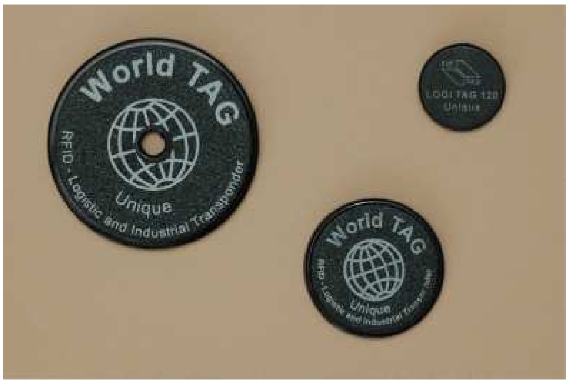

World TAG
RFID - Logistic and Industrial Transponder
Unique
LOGI TAG 120
Unique
World TAG
RFID - Logistic and Industrial Transponder
Unique

Figure 5.2: Logistic and industry

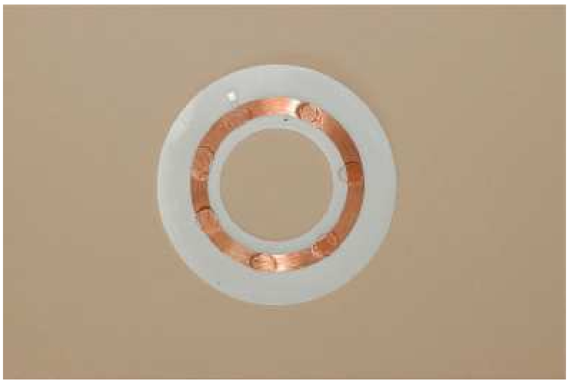


Figure 5.3: CDD label

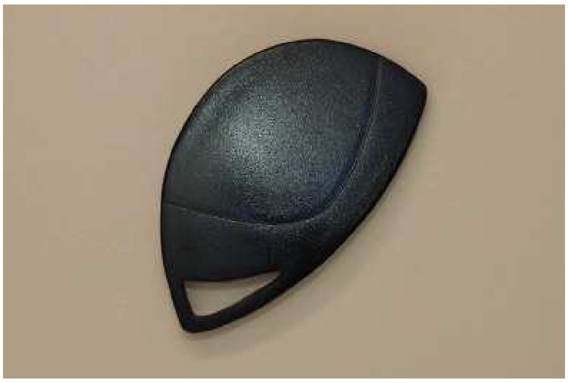


Figure 5.4: Key fobs

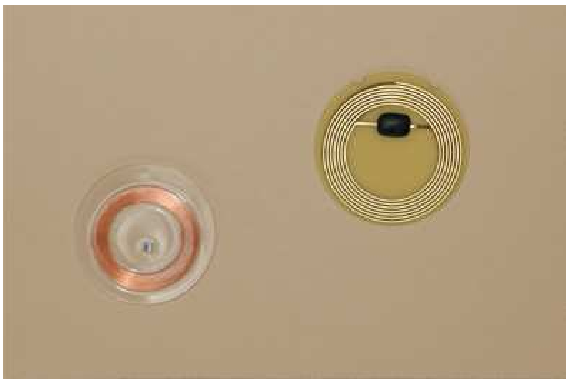


Figure 5.5: Naked (left) and disc (right) tags

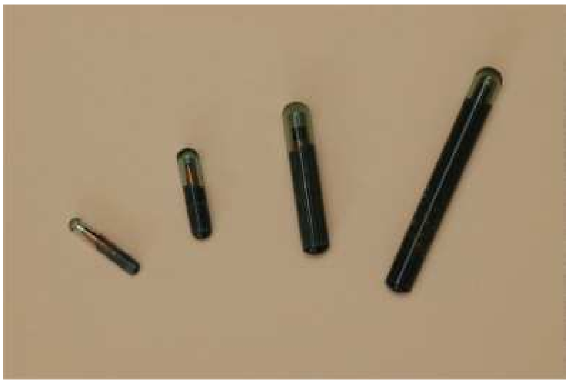


Figure 5.6: Livestock or pet identification

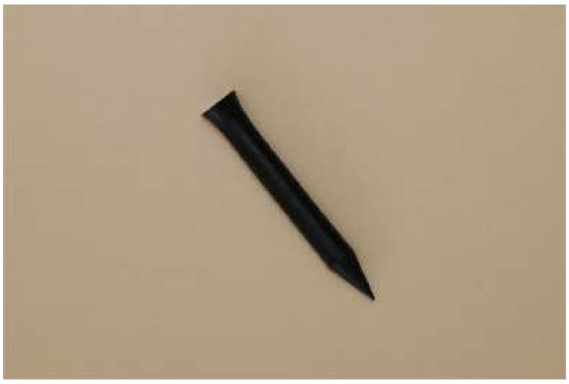


Figure 5.7: Nail tag

5.3 Tags Characteristics

5.3.1 Power

The power required to communicate and to carry out computations can be either internal or external to the tags. *Passive* tags do not possess any internal energy source. They obtain energy from the reader's electromagnetic field. There also exist *semi-passive* tags where a battery is used for internal calculations. However, the energy required for transmission still comes from the reader's electromagnetic field. Lastly, there exist *active* tags, i.e., tags that have a battery that is used for internal calculations and for transmission. Active and semi-passive tags are more expensive than passive tags, are generally more cumbersome, and have a limited lifetime. Hereafter, we only consider passive tags.

5.3.2 Communication Range

The communication range between the reader and the tag depends on several parameters. It of course depends on the frequency used, but also on the transmission power, that is standardized in Europe by the ETSI EN 300-330, EN 300-220, EN 300-440, and EN 300-328 standards. Practically, the RFID systems communication range using low frequency (LF) and high frequency (HF) bands is a few decimeters, even centimeters, while systems using the ultra high frequency (UHF) band can communicate up to several meters. The ranges mentioned here are obviously the limits that one can practically observe using standard material and respecting the regulations. However, it is important to underline that using specific antennas and transmission powers above the legal limits, we can largely surpass these limits. Lastly, let us highlight that the ranges mentioned here correspond to the maximum distance between a reader and a tag, but the information sent by a reader (*forward channel*) can be captured at a distance far superior than that sent by a tag (*backward channel*).

5.3.3 Memory

All tags contain a minimum number of memory bits to store their identifier (between 32 and 128 bits). Depending on the targeted application, tags can have ROM, EEPROM, RAM or SRAM. The electronic anti-theft devices (EAS, Electronic Article Surveillance) that can be found on many items, require only one bit (enabled EAS / not enabled EAS). One may consider such devices as RFID tags. However, they do not allow object identification, only its detection. Thus we are inclined not to include them in our RFID terminology.

5.3.4 Computation Capabilities

Once again, it is hard to establish a tag classification depending on their computational capabilities. Certain tags have no computational capabilities and thus only have a simple memory that can be remotely accessed. In this work, such tags do not interest us since the cryptographic protocols to be implemented between the tag and the reader are simply non-existent. Other tags that have a simple design and thus are relatively inexpensive, can perform certain XOR and AND operations. Next, we have more evolved tags that have a few thousand logical gates and are thus capable of integrating a symmetric encryption or a hash function. Lastly, we can think of much more evolved tags that could use asymmetric cryptography. However, these types of tags cannot be classified as "low cost" and thus do not follow the trend that focuses on tags that cost a few Euro cents.

As previously stated, there is no clear demarcation between the different tag types. But if technology does not allow such a classification, then current cryptographic researches more or less allows it. Thus, currently we can distinguish between researches that involve tags capable of performing certain basic operations (e.g., [112]) and researches that assume a symmetric cryptographic function being usable in a tag (e.g., [181]). In fact, this demarcation is an *a posteriori* one, done after analyzing papers published in the RFID systems security domain. This classification overlaps a bit with the one proposed by EPCGlobal, presented in Section 5.3.8.

5.3.5 Tamper-Resistance

Currently, tags' tamper-resistance is a controversial issue. The question to be asked is whether it would be possible to obtain data from a tag by carrying out a physical attack on it [4, 175].

The simplest tags send their (fixed) identifier when they interact with the reader (see Section 5.4). Thus it is not useful to carry out a physical attack to obtain the tag's data contents. More evolved tags that use challenge-response authentication protocols, can benefit from physical protection. However such protections would be too expensive and unrealistic for tags that only cost a few Euro cents. Thus, they are reserved for more sensitive applications, where the tag costs can reach a few dollars.

Does this signify that all security measures are impossible for low cost tags? The answer is no because, just like in all practical risk analysis, we should not consider the attack's feasibility, but its cost compared to its gain. For example, storing an identical cryptographic key in all tags of a system seems a bad approach since compromising a single tag would be sufficient to compromise the entire system. Using a different key for each tag seems a more reasonable approach.

In the majority of existing applications, obtaining a tag to attack the system is an easy thing. This is the case, for example, in libraries. However, the fact that a tag is not tamper-resistant can be counter balanced by the fact that in certain cases, it is hard for the adversary to gain access to a tag. This is the case, for example, in subdermal tags. A less extreme example is the use of electronic bracelets to localize people in a closed space, for example in an amusement park [152]. Indeed, the tag's contents can be reinitialized each time it is sold/returned by a visitor. On one hand, a visitor's aggression towards getting a bracelet is unlikely and can be detected. On the other hand an attack on a tag that is not carried by a visitor is of no purpose to the adversary, provided it was correctly reinitialized.

5.3.6 Physical Characteristics

Usually, it is the antenna and not the microcircuit that requires the most space in the tag (see for example Figure 5.3 and Figure 5.5). The antenna size mainly depends on the communication range and the frequency used (see Section 5.3.7). The smallest tag, as of today, is the μ-tag produced by Hitachi [169], that only measures 0.4 millimeters including the antenna (an external antenna can also be used).

Note that the term "tag" is generally not used to refer to the electronic device itself, but rather to the electronic device including its "package". For example, the μ-tag is not meant to be used on its own, but to be incorporated in another device, for example a card. Thus, we find several types of packages, such as glass tubes, key fobs, nails, etc. (see page 61). The small size of these devices is the reason why their presence is often ignored.

5.3.7 Frequency Band

There exist several frequency bands that can be used for RFID. The frequency band choice depends on the targeted application as it significantly influences the communication characteristics between the reader and the tag (transfer rate, communication range, communication degradation across the medium, etc.). It also depends on the regulations put in place for the region where the system will be deployed since it should not interfere with other systems such as the television, police mobile radio, marine, etc. Table 5.1 summarizes the four main frequency ranges used for RFID: 125–134 kHz (LF), 13.553–13.567 MHz (HF), 860–960 MHz (UHF), and 2.4000–2.4835 GHz (UHF). The other frequencies that are also used sometimes are 6.765–6.795 MHz, 7.400–8.800 MHz (used for EAS), 26.957–27.283 MHz, or 5.725–5.875 GHz.

Frequency range	Examples of applications
125–134 kHz (LF)	Pet identification, car keylocks, livestock tracking
13.553–13.567 MHz (HF)	Smart cards, library books, clothing identification
860–960 MHz (UHF)	Supply chain tracking
2.4000–2.4835 GHz (UHF)	Highway toll, vehicle fleet identification

Table 5.1: Main frequency ranges used in RFID

Tags using frequencies 13.553–13.567 MHz and 860–960 MHz are currently in the media's spotlight for many reasons, among them their adaptability to stock management. EPCGlobal has also focused its attention on these frequencies. However, low frequency tags have experienced a remarkable growth in terms of sales. Compared to the HF and UHF frequencies, with LF frequencies, there is less heat loss in the water. Thus LF frequencies are often preferred in biomedical applications.

Usually, higher the frequency, smaller is the antenna. However, this reasoning is only valid for the UHF frequency band, where the communication method used is backscatter coupling. In LF or HF bands, the communication method used is charge modulation by inductive coupling. In this case, not only do the size of the spirals play a role, but also the number of spirals. Thus a trade-off between the two is required. Thus, a tag designed for the 135 kHz frequency could be less cumbersome than a tag designed for the 2.45 GHz frequency.

The μ-tag previously mentioned, uses the 2.45 GHz frequency.

5.3.8 Standards

In the world of radio frequency identification, it is sometimes difficult not to lose oneself in the constellation of standards, either published or in the process of being elaborated. In this field, two main families stand out: ISO (International Organization for Standardization [105]) standards and EPC (Electronic Product Code [67]) standards.

In the ISO family, the key standards are 14443, 15693 and 18000, which deal with the communication protocols between tags and readers. Procedures have been published (ISO 10373 and 18047) to verify the conformity of a device with these standards. Procedures for measuring the efficiency of a device (ISO 18046) have also been standardized. Independent of the actual technology, the data organization on a tag is described in standards 15961, 15962, 15963 and 19789. Finally, some standards are specific to a given application, for example 17358 and 17363/7 for supply chains and 11785 for animal identification.

The EPC standards were established by EPCGlobal, a non profit organization made up of several companies and academics. Its aim is to standardize and promote very low cost RFID technology, with the goal of integrating it into supply chains. It is the Auto-ID Center, a research project that took place in 1999 at MIT, which was split in 2003, that gave birth to EPCGlobal and to Auto-ID Labs. Auto-ID Labs constitute a research center spread over seven universities. EPCGlobal distinguishes several classes of tags according to their function. Class 1 corresponds to the most simple tags, which have only a unique identifier for the tag by default, a code that allows the identification of the product to which it is attached and a function permitting the definitive destruction of the tag. Class 2 offers more memory and allows authentication functions to be carried out. Class 3 corresponds to semi-passive tags and finally class 4 corresponds to active tags, which can potentially communicate with each other. Today, only class 1 for ultra high frequency (860-960 MHz) is standardized. Ratified in December 2004, this standard, called “Class 1 Gen 2”, principally rests on the first generation specification of class 1 and of class 0 (which has disappeared from the nomenclature). In June 2005, the ISO confirmed its wish to integrate the EPCGlobal Class 1 Gen 2 standard into its norm 18000-6.

In order to illustrate the tags’ capacities, we supply in Table 5.2 a few examples of commercial RFID tags.

Tag	Frequency	Distance	Data rate	Unique ID	Memory R/W
(model)		(meters)	(kbits/s)	(bits)	(bits)
Hitag 1	LF	1.5	4	32	2048
Hitag S	LF	2.0	8	32	32, 256, 2048
ICode-1 SL1	HF	1.5	26.5	64	512
UCode EPC G2	UHF	7.0	640	32	512

Table 5.2: Philips’ RFID tags (source: http://www.semiconductors.philips.com)

5.3.9 Coupling and Data Transfer

To conclude this technical overview of the tag characteristics, we give a rough approach of the electronic aspect of the communication between a reader and a tag. There are two basic techniques for acquiring information from RFID tags: *inductive coupling* and *backscatter coupling*. The former is used with frequencies less than 400 MHz, in which case, the magnetic component of the electromagnetic field is dominant. Backscatter coupling is used with higher frequencies, where the electrical component of the electromagnetic field is the dominant component.

In inductively coupled systems (see Figure 5.8), the reader's antenna coil generates an electromagnetic field that penetrates the antenna coil of the tag. By induction, a voltage is generated, which is rectified thanks to the diode, and supplied to the capacitor. When the capacitor is sufficiently loaded, the chip is activated and it outputs a signal, e.g., its identifier. This signal activates and deactivates a field effect transistor which modulates the signal returned to the reader.

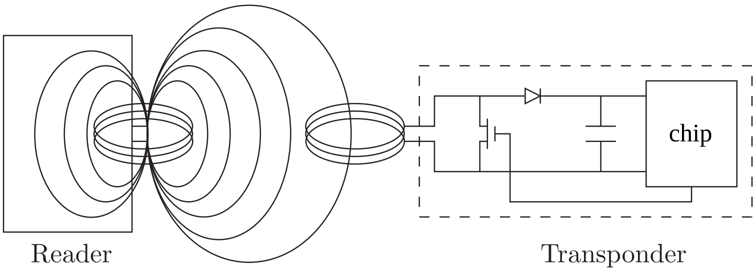


Figure 5.8: Inductive coupling

In backscatter coupling systems (see Figure 5.9), a portion of the incoming energy is reflected by the tag's antenna and re-radiated outwards. In order to transmit data, the signal outputted by the chip activates or deactivates a transistor that modulates the energy reflected by the antenna.

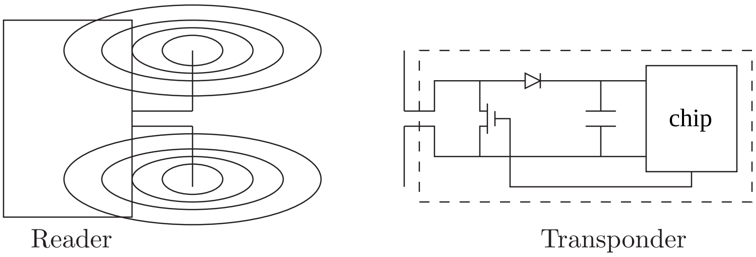


Figure 5.9: Backscatter coupling

5.4 Identification and Authentication Protocols

5.4.1 Identification vs Authentication

A major issue when designing a protocol is defining its purpose. This issue, as trivial as it may seem, does not have any immediate solution when we speak of RFID. Determining if it is an identification or an authentication protocol that is required, and knowing what each term signifies remains a confusing issue and has provoked several misunderstandings. We provide our opinion here.

When we analyze RFID applications, it is possible to think of two large application categories: those whose goal is to provide security to a system, and those whose goal is to provide functionality, in other term users' convenience, without any security concerns. In the first category, we find applications such as access control (badge to access a restricted area, car ignition key, etc.) or prevention against counterfeits. In the second category, we find RFID being used to improve supply chains, to facilitate location and stocking of books in libraries, to localize people in amusement parks, to facilitate sorting of recyclable material, to improve consumer services in shops, to count cattle, etc. Clearly, the security aspect is not the driving force in this category.

However, let us note that sometimes, deployment of an application creates a security problem that was previously non-existent. For example, using RFID tags in place of barcodes in shops *a priori* does not pose any security problems since an RFID tag is not less secure than a barcode (duplicable by a simple photocopy). However, since we are always ambitious to go further and further, removing the cashier poses a security problem. Indeed, there is no more manual control and we would ask the RFID tag to do more than what was expected of the barcodes, that is avoid counterfeits. This example can thus be classified under the first category, the one that requires security.

To summarize, in the first category, the objective is to ensure the tag authentication. In the second category, the objective is simply to obtain the tag's identity, without any proof being required. Below, in the RFID framework, we define the concepts of *authentication*, *mutual authentication*, and *identification*.

Definition 10 (authentication protocol). *An authentication protocol allows a reader to be convinced of the identity of a queried tag. Conversely, it can allow a tag to be convinced of the identity of a querying reader. If both properties are ensured, we speak of* mutual *authentication.*

Definition 11 (identification protocol). *An identification protocol allows a reader to obtain the identity of a queried tag, but no proof is required.*

To illustrate our identification concept, we provide an example that only ensures identification but can be later on integrated in a solution that ensures authentication. Thus, biometric authentication systems, for example those that rely on fingerprint recognition, are currently more and more used in physical access control. Such a system can be deployed for example, to control access to a building. It is also possible to use such a solution conjointly with a token, for example a smart card, in which the fingerprint of the person to be authenticated would be stored. Another solution is to store in a centralized database system, the fingerprints of all authorized personnel that can enter the building. In this case, when a person presents himself, the system does not know his identity and must perform an exhaustive search among all the stored fingerprints. If the number of stored fingerprints is large, this process could take a few seconds and forces the person to wait in front of the door.

To solve this problem, one solution is to use a token, not for storing the fingerprint, but for storing the person's identity. When inserting this token in the reader, the system immediately identifies which fingerprint it must test, thus avoiding the exhaustive search on all stored fingerprints. Using a traditional smart card reader is not in the current trend. On one hand, such a reader requires that the person take his smart card out of his pocket; on the other hand managing such a large number of readers can be problematic at times, not only due to breakdowns, but also due to vandalism. The solution envisaged by certain companies is to use an RFID system whose only objective is to provide to the biometric system, the identity of the person that presents himself. Thus, in this case the tag has nothing to contribute to the security and is only for convenience, it avoids waiting in front of the door while the system performs its search.

5.4.2 Protocols

We have seen that both authentication and identification protocols are useful in practice. Of course, authentication protocols also ensure identification, but it seems obvious that identification protocols may be cheaper than authentication protocols.

The basic identification protocol that is used in today's applications (e.g., pet identification) is depicted in Figure 5.10. It consists in a request from the reader to the tag and an answer from the tag to the reader, containing the identifier ID of the tag. If the tag is a legitimate one, then the system's database also contains ID. This protocol is simple, sound, but we will see in Chapter 6 that it brings out privacy issues.

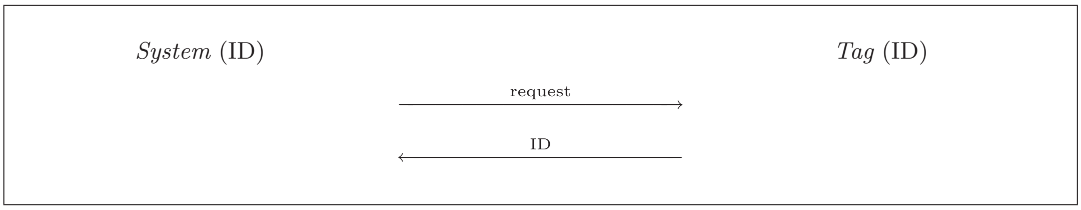


Figure 5.10: Basic identification scheme

The basic authentication protocol that is used in today's applications (e.g., car ignition key) is depicted in Figure 5.11. It consists in a common challenge-response protocol: the reader sends a nonce a to the tag and the latter answers ID and $F(\text{ID}, a)$ where F is a given function, e.g., an encryption function. Again, this protocol endangers the privacy of the tag's bearer as we will see in Chapter 6.

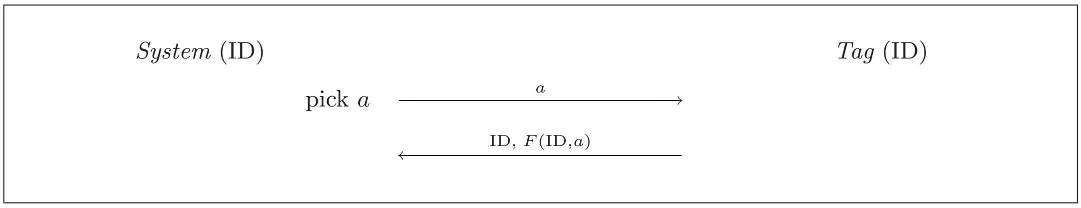


Figure 5.11: Basic challenge-response authentication scheme

Currently, even though RFID gains more amplitude, many people think that we could do everything with low cost tags. The threat that lurks is misusing the solutions for applications that require security, for example, using an identification protocol for authentication purposes, just for reducing the tag price. That is precisely what happened at the MIT, where Agrawal, Bhargava, Chandrasekhar, Dahya, and Zamfirescu have shown that the new MIT ID Cards that contain RFID tags only send their identifier "in clear" to authenticate their owners. As Agrawal *et al.* said [1]: "If you are counting cattle, the cattle are not going to impersonate each other; there is no need for strong encryption. Members of the MIT population are not cattle; we need strong encryption."

Security and Privacy

CHAPTER SIX

Like all growing technologies, radio frequency identification brings along its share of security related problems. Threats in RFID systems can be classified in two categories. The first category, which includes common security threats, concerns those attacks that aim at wiping out the system's purpose. Such attacks can target to crash the system using, for example a denial of service attack, but it can also be just to impersonate a given tag. The second category is related to privacy: the problem is information leakage, as a tag may reveal data about an object or a person (passport, ID-card, medical dossier, etc.), and traceability. By "traceability" we mean an adversary is able to recognize a tag that she has already seen, at another time or in another place. In the near future, people will surely carry several tags on themselves, for example in their clothes, shoes, or even on their keys, that will serve as a type of digital fingerprint allowing themselves to be traced.

The defenders of RFID minimize the privacy problems while its detractors amplify it. More concretely, some companies had to renounce this technology after being boycotted by organizations that defend individuals' liberty. This shows that we need to seriously address this problem. Several palliative solutions already exist but they involve so many constraints on the tag holders that their use can be debated. Thus the solution lies in the conception of identification or authentication protocols that do no threaten the tag holder's privacy. The problem is that finding such a protocol is far from being an easy task, due to the weak resources available on tags.

In this chapter, we start by describing the threats against the RFID technology. Then, in Section 6.2, we discuss existing measures and general concepts of RFID protocols. Finally in Section 6.3, we show that privacy must not be treated in the "classical" theoretical model only, but must be considered in the communication model as a whole, right from the physical layer to the application layer.

6.1 Security Threats

In our opinion, the first paper which really opened the way to research devoted to security and privacy in RFID systems is the one of Sarma, Weis, and Engels [155]. Although informal, this paper lays down the security and privacy issues in RFID. Since then, many papers have dealt with these issues. A few reference papers are [26, 73, 113, 117, 136, 178]. For further views on the risk of using tags, we suggest the reader to have a look at some other papers, for example [15, 19, 72, 96, 112, 118, 122, 134, 156]. We also suggest the reading of the master thesis of Yang [111] and Hjorth [103], and the quite innovative master thesis of Weis [176]. Finally, several interesting books related to this topic have been published: [78] that gives a thorough introduction to RFID; [89], which is more dedicated to security and privacy; and [3] that focuses on privacy violation.

6.1.1 Impersonation and Relay Attack

Speaking of tags' impersonation does not make sense when dealing with identification protocols. Indeed, as seen in the previous chapter, identification protocols do not require an identity proof. However, being resistant to impersonation is one of the requirements of authentication protocols, along with completeness:

1. *Completeness*: the reader always accept the legitimate tag.
2. *Impersonation*: the probability is negligible that any adversary other than the tag, carrying out the protocol playing the role of the tag, can cause the legitimate reader to complete and accept the tag's identity.

We emphasize that the theoretical properties given above are not sufficient to ensure security in practice. Indeed, the fact that the tags can be queried without the agreement of their carriers opens the way to so-called *relay attacks*. It consists, for a pirate, of making the reader believe that the tag is present in its electromagnetic field, while, in fact, it is not. For that, the adversary uses two devices: one simulates the legitimate tag in the field of the reader while the other one simulates a legitimate reader close to the tag (see Figure 6.1).

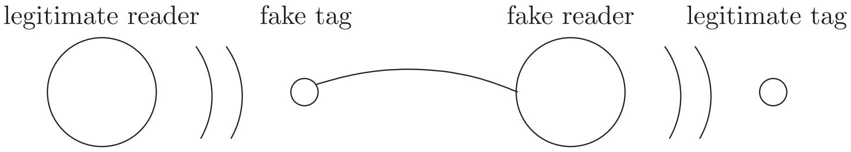


Figure 6.1: Relay attack

A relay attack can be passive, meaning that the adversary plays the role of an extension cord as in the example above, or it can be active, enabling the adversary to perform more sophisticated man-in-the-middle attacks.

Relay attacks are also known as *mafia fraud*, which have been introduced in 1987 by Desmedt, Goutier, and Bengio [62]. The term "mafia fraud" comes from an interview given by Shamir to Gleick, where he said about his protocol [76], "I can go to a Mafia-owned store a million successive times and they still will not be able to misrepresent themselves as me".

Technical measures to avoid this sort of attack rely on distance-bounding. The idea, initially suggested in 1990 by Beth and Desmedt [38] and then studied thoroughly by Brands and Chaum [48], consists in computing the delay time of a one-bit exchange between the prover and the verifier. If the delay time is lower than a given threshold, then the verifier considers with probability 1/2 that no attack occurs in this round. Indeed, an attack that consists in answering randomly succeeds with probability 1/2. Consequently, the one-bit exchange is repeated until the probability of detecting an occurring attack is high enough. Recently, several other distance bounding protocols have been proposed, for example [51, 52, 157].

In the RFID framework, Hancke and Kuhn [98] proposed such a distance-bounding protocol. The sketch of their protocol is depicted in Figure 6.2. Let us note that Hancke and Kuhn's protocol has two disadvantages.

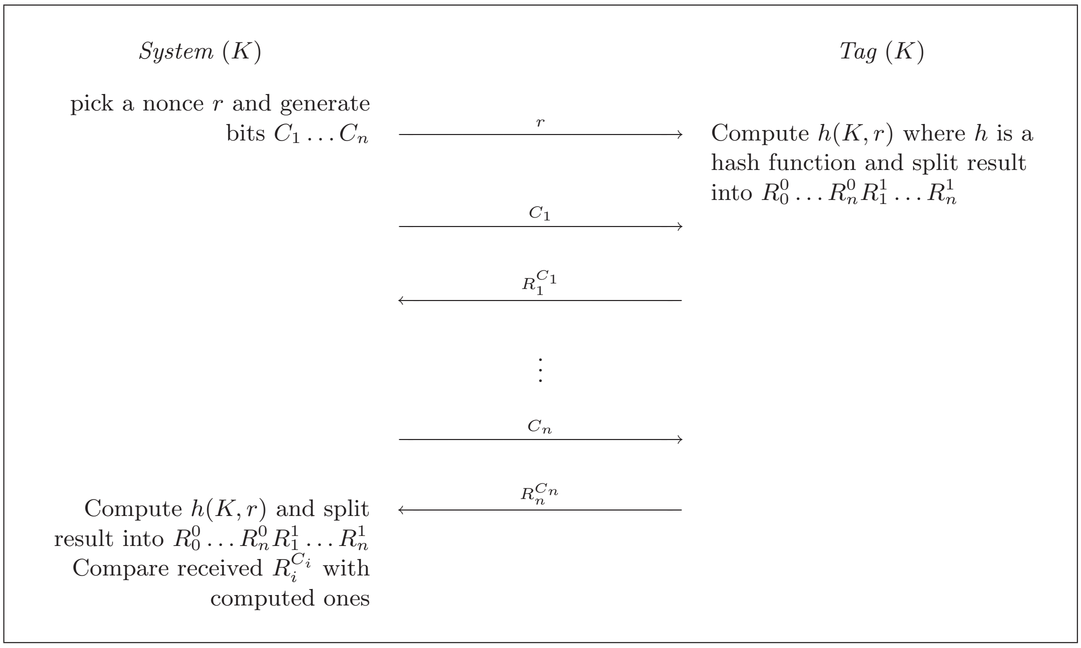

System (K)
Tag (K)
pick a nonce r and generate bits $C_1 \dots C_n$
r
Compute $h(K,r)$ where h is a hash function and split result into $R^0_0 \dots R^0_n R^1_1 \dots R^1_n$
C_1
$R_1^{C_1}$
$\vdots$
C_n
$R_n^{C_n}$
Compute $h(K,r)$ and split result into $R^0_0 \dots R^0_n R^1_1 \dots R^1_n$
Compare received $R_i^{C_i}$ with computed ones

Figure 6.2: Hancke and Kuhn's distance-bounding protocol

Firstly, the success probability for an adversary to impersonate a legitimate prover is $(3/4)^n$ and not $(1/2)^n$. This is because: the adversary can query the tag just after receiving the nonce r with any C_i (for example all the C_is equal to 0), thus obtaining half of the R_is. Since the critical challenges exchange phase has not yet started, the adversary has enough time to query the tag n times. Then, the adversary interacts with the verifier: in one out of two cases, the adversary has the right answer; and in one out of two cases she must randomly respond. This brings us to the $(3/4)^n$ probability mentioned above.

The second disadvantage of [98] is we have to prevent the adversary being able to query the tag $2n$ times with the same nonce r. In this case, the adversary would obtain the $2n$ bits $R^0_0 \dots R^0_n R^1_1 \dots R^1_n$ and thus her relay attack would succeed with probability 1. Two solutions are considered to avoid this attack. Either we have to suppose that the adversary does not have the time to carry out $2n$ requests between the nonce exchange phase and the critical challenges exchange phase, or the $R^0_0 \dots R^0_n R^1_1 \dots R^1_n$ bits must not be generated uniquely from a value provided by the reader, but also from a value generated by the tag. Thus one would have to add an exchange in the protocol to inform the reader of the value chosen by the tag.

Relay attacks pose a threat to certain RFID systems, particularly to those whose objective is to ensure access control. The danger arises from the fact that the victim would be unaware of the attack as his tag responds to the reader's request.

For example, to start a vehicle, an adversary equipped with a portable computer places herself near to the vehicle while her accomplice places herself at the vehicle owner's side. The car's reader sends a challenge to the adversary, who transmits it to her accomplice. This challenge is sent to the victim's key which responds correctly. This response is resent to the adversary who finally submits it to the vehicle's reader. The electronic protection of the vehicle is thus breached without the adversary having the key.

Finally, Fishkin, Roy, and Jiang [80] notice that malicious readers are usually distant from the targeted tag, while legitimate readers are closer. So, they suggest a mechanism that allows tags to estimate reader distance. Their mechanism is a low-level technique based on the analysis of the received signal. The authors explain that a variation on the signal strength analysis correlates tag distance to received energy.

6.1.2 Information Leakage

The information leakage problem emerges when data sent by the tag reveals information intrinsic to the marked object. In the framework of a library, for example, the information openly communicated could be the title of the work, which may not please some readers. More worryingly, marked pharmaceutical products, as advocated by the Food and Drug Administration in the United States, could reveal a person's pathology. For example, an employer or an insurer could find out which medicines a person is taking and thus work out his state of health. It is particularly important to take this aspect into account for the more advanced applications where personal information is contained in the transponder (electronic dossier, biometric passport, etc.).

In the United States, the arrival of the passport equipped with an electronic tag (see [115]), originally anticipated for August 2005, has been postponed in order to reconsider the security issues.

In California, Senate Bill 682, introduced by Senator Simitian, plans to restrict the use of RFID-based identification cards. In its initial version, the pending act was quite restrictive:

> *"The act would prohibit identity documents created, mandated, or issued by various public entities from containing a contactless integrated circuit or other device that can broadcast personal information or enable personal information to be scanned remotely."*

Since then, this bill has been amended several times by both the Senate and the Assembly. In its last published amended version (August 15, 2005), the bill states:

> *"Except as provided in subdivisions (b) and (c), all identification documents created, mandated, purchased, or issued by a state, country, or municipal government, or subdivision or agency thereof that use radio waves to transmit personal information or to enable information to be read remotely shall meet [certain] requirements."*

The suggested technical requirements can be summarized as follows:

1. The identification document shall not transmit or enable the remote reading of any personal information other than a unique personal identifier number using radio waves.
2. The identification document shall implement mutual authentication.
3. The identification document shall implement key establishment.
4. The identification document shall implement strong encryption.
5. The identification document shall implement at least one of the following privacy safeguards, in order to ensure that the holder of the identification document affirmatively consents to each reading:
   (a) An access control protocol requiring the optical or other non radio frequency reading of information from the identification document prior to each transmission or broadcast of data using radio waves.
   (b) A shield device that can prevent any communication of data using radio waves between the identification document and any reader under any circumstances.
   (c) A data-carrying device that is normally not remotely readable under any circumstances except while being temporarily switched on or otherwise intentionally activated by a person in physical possession of the identification document.

6.1.3 Foretaste of Traceability

Electronic tags are not cut out to contain or transmit large quantities of information. When a database is present in the system, the tag may only send a simple identifier, which only people having access to the database can link up to the corresponding object. However, even if an identifier does not allow obtaining information about the object itself, it allows us to trace it, that is to say, to recognize the object in different places and/or at different times. Thus we can know when a person passed through a given place, for example to work out his time of arrival or departure from his place of work. We could also piece together, from several readers, the path taken by a person, for example in a store or shopping mall.

Other technologies also permit the tracking of people, e.g., video surveillance, GSM, Bluetooth. However, RFID tags permit everybody to track people using low cost equipment. This is strengthened by the fact that tags cannot be switched off, they can easily be hidden, their lifespan is not limited, and analyzing the collected data can be efficiently automated.

Advocates of this technology arduously refute the argument that electronic tags put respect for privacy in peril. A maximum reading distance reduced to a few decimeters is the principal defense argument. In the framework of mass marketing, the physical or electronic destruction of the tags during their passage through the checkout is also an approach envisaged to make this technology more acceptable. From the opposition's point of view, the short reading distance is not a relevant security argument. Indeed, by using a more efficient antenna and a stronger power, it is possible to go beyond the presupposed limit. Moreover, there are many cases where an adversary can get close enough to her victim to read his electronic tags: on public transport, waiting in line, etc. Finally, the last argument of any substance is that the trend is not towards HF systems that have a short communication distance, but towards UHF systems, where the communication distance is a few meters.

Many voices have spoken out through several boycott campaigns, aimed at reversing the trend for omnipresent electronic tags in everyday life. CASPIAN (Consumers Against Supermarket Privacy Invasion and Numbering), an organization for the defense of individual liberties led by Albrecht, have called for a moratorium on the use of electronic tags for individual items. Wal-Mart, Procter & Gamble, Gillette and Tesco, who are not in favor of the moratorium, have had to face particularly virulent boycott campaigns. In Germany, it is the Metro group who has suffered such attacks at the hands of FoeBud, a militant human rights organization. They issued a "Big Brother Award" to Metro for their experimental shop in Rheinberg, where customers' loyalty cards unknowingly contained electronic tags.

Consequently, even if the majority of people are hardly worried about the threat that RFID poses to privacy, the weight of organizations such as CASPIAN and FoeBud is sufficiently great to slow down the deployment of the RFID market. The problem of malicious traceability of individuals is thus an aspect of RFID that has to be taken into account and treated seriously.

6.2 Measures Protecting Privacy

6.2.1 Palliative Techniques

The first technique that can be used to protect the tag's holders is to kill the tags. It is well-suited for example to supply chains: when the tag reaches the end of the chain, e.g., during the shop checkout, it is killed. The technique is effective but has several drawbacks: the management of the keys is complex in case of keyed kill-command, the tag can no longer be used afterwards, and there is no confirmation of successful disablement.

To avoid these drawbacks, Karjoth and Moskowitz [119] suggest the *clipped tags*. These devices are tags whose antenna can be physically separated from the chip. The bearer can carry out this operation by himself, and reactivation of the tag can only be done intentionally.

A less radical method consists of preventing the tag from hearing the request by enclosing it in a Faraday cage. This solution is only suitable for a few precise applications, e.g., money wallets or passports, but is not for general use: animal identification is an example of an application that could not benefit from this technique. Future US passports including an RFID tag may contain a thin radio shield in their cover, preventing the tags from being read when the passports are closed.

The third technique consists of preventing the reader from understanding the reply. The best illustration of this technique is surely the *blocker tag* [114, 117] that aims at preventing a reader from determining which tags are present in its environment. Roughly speaking, the blocker tag relies on the tree walking protocol (see Section 6.3.1) and simulates the full spectrum of possible identifiers.

Another technical approach consists in requiring an optical contact with the object prior to querying its tag remotely. Such an approach is very restrictive but is suited to certain applications. It is an approach suggested by Juels and Pappu [116] to protect banknotes. It is also the approach that may be used to protect the privacy of the US passport bearers: the officer must swipe the passport through an optical reader to get the key that gives access to the electronic data inside the tag. This technique is also advised in Senate Bill 682 presented in Section 6.1.2.

Finally, a rather different and complementary approach is that of Garfinkel, who elaborates the so-called "RFID Bill of Rights" [87, 88], which outlines the fundamental rights of the tag's bearers. Garfinkel claims:

- The right to know whether products contain RFID tags.
- The right to have RFID tags removed or deactivated when they purchase products.
- The right to use RFID-enabled services without RFID tags.
- The right to access an RFID tag's stored data.
- The right to know when, where and why the tags are being read.

In the same vein, Albrecht proposed the "RFID Right to Know Act of 2003", which requires that commodities containing RFID tags bear labels stating that fact.

Even though the methods presented here are efficient, they have many constraints that render them inconvenient. Thus the objective is to conceive tags querying protocols that protect the holders' privacy without imposing any constraints on them.

6.2.2 Protocols Resistant to Traceability

In Chapter 5 we have seen a sketch of the basic RFID protocols, be it identification or authentication. Unfortunately, these two sketches of protocols do not protect privacy at all since the tag identifier is sent in clear to the reader.

The naive idea in which the tag encrypts its identifier before sending it, with a key it shares with the reader creates more problems than it solves. Indeed, if the same key is used by all tags then obvious security problems arise in the case where the adversary can obtain the tag's contents. If a different key is used for each tag, then we have to consider two cases. In the first case, the encryption is deterministic, that is, the tag always sends back the same value, but with this, the traceability problem is not solved, as the encrypted object becomes the "marker" that can be used to trace the tag. In the second case, the encryption is randomized, which poses a complexity problem, as the reader has to test all the keys present in its database to find the one that matches the queried tag. This method, which is the only current one that ensures privacy with a realistic adversary model, is in fact a challenge-response where the verifier does not know the prover's identity. This is a fundamental point that differs from traditional cryptographic assumptions, where we generally suppose that each entity participating in the communication protocol knows the identity of the other entities participating in the same protocol.

Of course, a public-key *encryption* scheme could easily solve this problem: the prover encrypts his identity with the public key of the verifier. Thus, no eavesdropper is able to identify the prover. We stress that a public-key *authentication* scheme is not better than a symmetric authentication scheme in the following sense: in both cases, the system must test all the keys present in its database in order to identify the tag. A public-key authentication scheme may possibly reduce the computation on the tag's side (e.g., [92]) but computation on the system's side may become unmanageable when all keys must be tested. Anyway, public-key cryptography is too heavy to be implemented within low cost tags. Thus, we only deal with tags capable of performing XOR and AND operations, and tags capable of using symmetric cryptography, as seen in Section 5.3.4.

Thus, to avoid traceability, the adversary should not be able to tell the difference between the information sent by the tag and a random value. Also, the information sent by the tag should only be used once. In other words, each time the tag is queried, it sends a new random pseudonym to the reader, and only the latter knows that all these pseudonyms belong to the same person. The pseudonym sent by the tag is either the tag identifier, which is then refreshed at each new identification, or is the encrypted version of the identifier; in the latter case, the identifier contained in the tag can remain unchanged, but the encryption must be randomized. What differentiates the existing protocols is the way in which information sent by the tag, i.e., the pseudonyms, is refreshed. We can classify the protocols into two categories, those where the information refreshment is reader-aided and those where the information is refreshed by the tag itself, without the reader's help. Tags that are unable to use symmetric cryptography fall in the former category while more powerful tags usually fall in the latter one.

Protocols Based on Reader-Aided ID-Refreshment

In the case where the reader participates in refreshing information sent by the tag, the RFID protocol is usually a 3-moves protocol. As shown in Figure 6.3, the reader first sends a request to the tag; then the tag replies by sending an information that allows its identification, that is, its pseudonyms; finally the reader sends data that allows the tag to refresh the information it would send during the next identification.

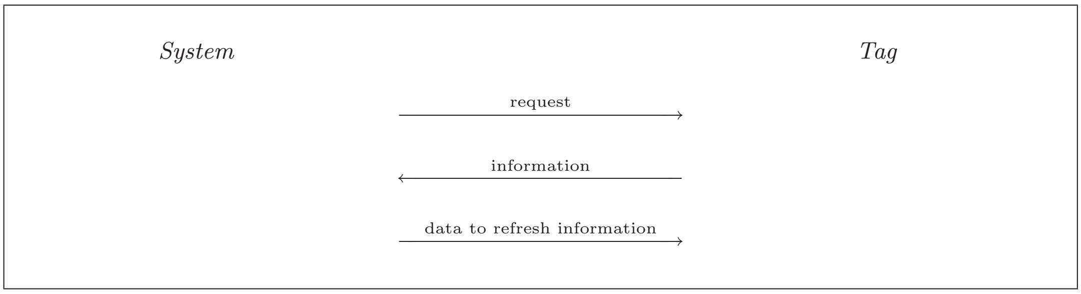


Figure 6.3: Sketch of an identification protocol

In this type of protocol, the difficulty lies in ensuring that the reader correctly carries out its work, that is, the sent data should allow proper refreshing of the information sent by the tag. If the adversary is active and not constrained on the number of successive requests that she can send to the tag, then she will eventually be able to trace the tag or ensure that the system itself can no longer identify it.

Thus from a traceability point of view, these protocols, for example, [93, 102, 112, 116, 153], are only secure when considering a weak adversary model. When designing such protocols, the challenge is thus to obtain the strongest possible adversary model, that is, to render the adversary's life hard. We have analyzed several existing protocols and will show our results in Chapter 7.

Protocols Based on Self-Refreshment

Protocols that rely on the fact that tags refresh themselves their identifiers, for example, [73, 136], are usually 2-moves or possibly 3-moves protocols when there is a mutual authentication between the reader and the tag. These protocols usually use a hash or an encryption function on the tag's side and thus are not suitable for low cost tags. In [118], Juels and Weis propose a protocol based on self-refreshment that only uses XOR and AND operations. Unfortunately, as mentioned in Section 8.6, this protocol is not resistant to certain types of active attacks.

6.2.3 Adversary Model

Designing and analyzing RFID protocols is still a real challenge because no universal adversary model has been defined yet: up until now designs and attacks have been made in an ad hoc way.

Even though we are aware of concepts such as *chosen plaintext attacks* (CPA), *non-adaptive chosen ciphertext attacks* (CCA1), or *adaptive chosen ciphertext attacks* (CCA2) for confidentiality, and concepts such as *known-message attacks* (KMA) and *adaptive chosen message attacks* (CMA) for signature schemes, in radio frequency identification, the adversary's resources are defined in an ad hoc manner. Depending on the publications, the adversary is passive, active but limited in the number of successive queries to the tag [112], active but cannot modify the exchange between a legitimate reader and a tag [118], active but cannot tamper with the tags, or finally the only restriction of the adversary can be the reader's tamper-resistance [138].

The goal of the adversary is also not clearly defined. While a public key encryption scheme verifies, for example, the properties of *indistinguishability* (IND) or of *non-malleability* (NM), or that a signature scheme is resistant to *forgery* or to *total break*, the concept of *untraceability* (UNT) is not yet defined.

So, we define *untraceability* as follows: given a set of readings between tags and readers, an adversary must not be able to find any relation between any readings of a same tag or set of tags. Since tags are not tamper-resistant, *a priori*, an adversary may even obtain the data stored in the tags' memory in addition to the readings from the readers/tags. Thus, she might become capable of tracing the tags' past events, given their content. Therefore, we define *forward untraceability*: given a set of readings between tags and readers and given the fact that all information stored in the involved tags has been revealed at time t, the adversary must not be able to find any relation between any readings of a same tag or set of tags that occurred at time $t' \leq t$. *Privacy* also includes the fact that a tag must not reveal any information about the kind of item it is attached to. These rather informal definitions are needed to analyze the protocols given in Chapter 7. In Chapter 9, we provide a more formal but still ad hoc definition of untraceability in order to analyze Molnar and Wagner's protocol [136].

Formalizing these definitions is something that will have to be done in the near future. We have already proposed an outline of a formal adversary model [16], but this will require more modifications so as to perfectly suit RFID.

6.3 Traceability in the Lower Layers

The communication channels are usually devised using a layered approach, for example the OSI model [108]. By implementing a cryptographic protocol at a given layer, confidentiality, integrity, and authentication can be guaranteed independently of the lower layers characteristics, since the useful data are contained in the highest layer. With regards to traceability, the problem is very different, as we explain in [26]. Each layer can reveal information that can be used to trace a tag. Hence we have to prove that traceability is prevented at each layer. Thus, a protocol that is safe with regards to traceability in a classic adversary model dealing only with the application layer may not be safe in practice. This point is rarely taken into account in the protocols that are described in the literature. According to our knowledge, only Juels *et al.* [117], Molnar and Wagner [136], and Weis [176] have tackled this problem.

Below, we describe the layered communication model that is used in RFID. This model, that is compatible with the ISO standard 18000-1 [106], is made of three layers, the application, the communication and the physical layers.

- The *application layer* handles the information defined by the user. This could be information about the tagged object (e.g., the title of a book) or more probably an identifier allowing the reader to extract the corresponding information from a database. The identification or authentication protocol is defined in this layer.

- The *communication layer* defines the way in which the readers and tags can communicate. Thus, the collision-avoidance protocol is found in this layer. This layer also contains an identifier that is used by the collision-avoidance protocol in order to singulate the tag (this identifier does not have to be the same as the one in the application layer). We use the term *singulation identifier* to denote such an identifier, or more simply *identifier* where there is no ambiguity with the identifier of the application layer.

- The *physical layer* defines the physical air interface, that is to say, frequency, transmission modulation, data encoding, timings and so on.

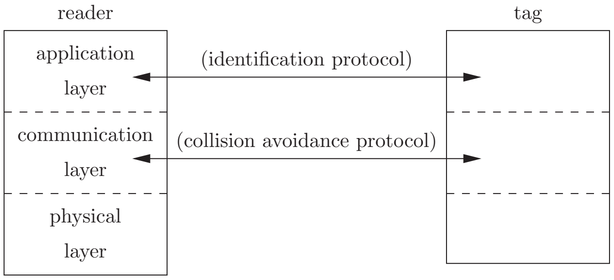


Figure 6.4: Communication model

In the next section, we provide some background on collision-avoidance protocols and later on, we address the traceability issue in the communication and physical layers.

6.3.1 Collision-Avoidance Protocols

Singulation

With several entities communicating on the same channel, we need to define rules to avoid collisions and therefore avoid information loss. This issue arises in RFID systems because when a reader broadcasts a request, all the tags in its field reply simultaneously, causing collisions. The rules required define the *collision-avoidance* protocol. Since tag's computational power is very limited and tags are unable to communicate with each other, the readers must deal with the collision-avoidance themselves. Usually, it consists of querying the tags until all singulation identifiers are obtained. We say that the reader performs the *singulation* of tags because it can then request them selectively, without collision, by indicating the identifier of the queried tag in its request. The collision-avoidance protocols that are used in current RFID systems are often (non-open source) proprietary algorithms. Therefore, obtaining information on them is quite hard. However, several open standards appear either from the EPC family [67] or from the ISO family [105].

There exist actually several collision-avoidance protocols, depending (in part) on the frequency used. EPC proposes standards for the frequencies 13.56 MHz and 860-930 MHz. ISO proposes standards from 18000-1 to 18000-6 where 18000-3 corresponds to frequency 13.56 MHz, and 18000-6 corresponds to frequency 860-960 MHz. We have two main classes of collision-avoidance protocols: the deterministic protocols and the probabilistic protocols. Usually, we use probabilistic protocols for systems using frequency 13.56 MHz, and deterministic protocols for systems using frequency 860-960 MHz.

Below, we describe these two classes.

Deterministic Protocols

Deterministic protocols rely on the fact that each tag has a unique identifier. If we want the singulation process to succeed, the identifiers must stay unchanged until the end of the process. In current tags, the identifiers are set by the tag manufacturer and written in the tag's ROM. We give an example of deterministic collision-avoidance protocol called *tree walking*.

Let us consider tags having a unique identifier of bit-length ℓ. All the possible identifiers can be visualized by a binary tree of depth ℓ. A node at depth d in this tree can be uniquely identified by a binary prefix $b_1 b_2 ... b_d$. The reader starts at the root of the tree and performs a recursive depth-first search. So, at node $b_1 b_2 ... b_d$, the reader queries all tags whose serial numbers bear this prefix, the others remain silent. The tags reply with the $d+1$-st bit of their serial number. If a collision occurs, the reader must explore both children of the prefix. When the algorithm reaches a leaf, it has detected a tag. The full output of the algorithm is a list of all tags within the field of the reader.

Probabilistic Protocols

Probabilistic protocols are usually based on a time-division multiple access protocol, called *Aloha*. We describe one of the variants of Aloha, namely the slotted Aloha (see Figure 6.5). In the slotted Aloha, access to the communication channel is split into time slots. In general, the number of slots n is chosen randomly by the reader, which informs the tags on how many slots they will have to answer to its singulation request. Each tag randomly chooses one slot among n and responds to the reader when its slot arrives. If n is not sufficiently large with regards to the number of tags that are present, then some collisions will occur. In order to recover the missing information, the reader queries the tags once more. It can mute the tags that have not brought out collisions (*switched-off* technique) by indicating their identifiers or the time slots during which they transmitted. Also, according to the number of collisions, it can choose a more appropriate n.

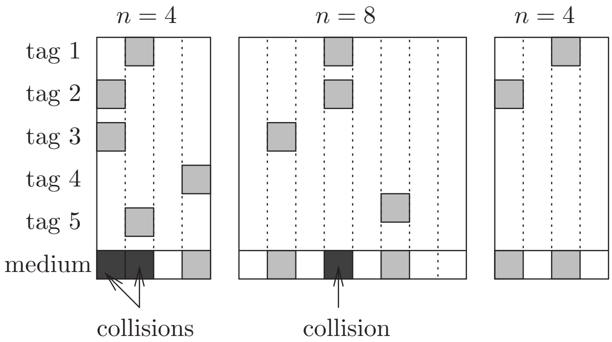


Figure 6.5: Slotted Aloha with the switched-off technique

Note that, the fact that each tag has a unique singulation identifier is not a fundamental requirement for Aloha, but is desirable for efficiency reasons [107]. Without using these identifiers, the exchange of information of the application layer is carried out during the singulation because the reader can no more communicate with the tag once the singulation process is completed. Note also that the singulation seems *atomic* from the tag's view: while a tag must reply to the reader several times when the tree walking is used, the tag can answer only once when no collision occurs with the Aloha protocol. In the case where the response brings out a collision, the reader restarts a new singulation process with a possibly larger n. On the other hand, if the switched-off technique is used, then the protocol is no more atomic. We detail this point in the next section.

6.3.2 Traceability Within the Communication layer

Threats Due to an Uncompleted Singulation Session

It is clear that deterministic collision-avoidance protocols relying on static identifiers give the adversary an easy way to track the tags. To avoid traceability, the identifiers would need to be dynamic. However if the identifier is modified during the singulation process, singulation becomes impossible. Thus we introduce the concept of *singulation session* as being the set of exchanges between a reader and a tag that are needed to singulate the latter. When the session does not finish, due to failures or attacks, we say that the session stays *open*.

Since the singulation identifier cannot be changed during a session, the idea, to avoid traceability, is to use an identifier that is different for each session. The fact that the tag can be tracked during a session is not really a problem due to the shortness of such a session. In practice, the notion of singulation session already exists informally because the readers usually send a signal at the beginning and at the end of a singulation. Unfortunately, there is no reason to trust the readers to correctly accomplish this task. In particular, a malicious reader can voluntarily keep a session open to track the tag thanks to the unchanged identifier. This attack cannot be avoided when the signals come from the reader and not from the tag itself.

Contrary to what we usually think, using a probabilistic protocol based on Aloha does not directly solve the traceability problem at the communication layer. Because, apart from the (inefficient) Aloha-based protocols that do not use the switched-off technique, the concept of singulation session is also needed with probabilistic singulation protocols. Indeed, after having queried the tags, the reader sends an acknowledgment (either to each tag or to all the tags) to indicate which tags should retransmit (either the reader acknowledges the identifiers of the tags it has successfully read, or it indicates the numbers of the slots where a collision occurred). In the case where identifiers are used, the fact that a singulation session stays open allows an adversary to track the tags. In the case where the acknowledgment does not contain identifiers but instead contains the numbers of the slots where a collision occurred, then an attack relying on these slots is also possible, as follows: an adversary who can communicate with her target tag sends it a (first) singulation request with the number of potential slots n. Assume the tag answers during the randomly chosen slot s_{target}. Since the tag is alone, the reader can easily link s_{target} to the targeted tag. The reader keeps the session opened. Later, when the adversary meets a set of tags potentially containing its target, she queries the tags again, indicating that only tags which transmitted during s_{target} must retransmit: if a tag retransmits, there is a high probability, depending on n and the number of tags in the set, that it is the adversary's target since another tag will respond to the second singulation request during s_{target} if and only if its last session stayed opened and it transmitted during s_{target}.

Whether we consider deterministic or probabilistic protocols, it is fundamental that singulation sessions cannot stay open. The tag should be able to detect such sessions and close them by itself. In other words, the signal needs to be internal to the tag. One may think that leaving the reader's field could be interpreted as such a signal, and therefore the tag refreshes its singulation identifier on leaving or entering a field. Unfortunately, even if this signal comes from the physical layer, it cannot be trusted: an adversary may maintain a constant field around the tag in order to track it. This attack is rather realistic in closed environments: for example in shopping centers, where merchants may have an interest in tracking the tags [15, 116].

Consequently, we suggest using an internal timeout to abort singulation sessions with abnormal duration. Thus, the tag starts the timeout when the singulation session begins (i.e., when it receives the first request of a singulation session). When the timeout expires, the current session is considered as aborted.

Implementation of such a timeout strongly depends on the practical system, e.g., the timeout could be a capacitor. When the tag receives the first request of a singulation session, it generates a fresh identifier and loads its capacitor. Then, each time it is queried (such that the request is not the first one of a session), it checks whether its capacitor is empty. If this is the case, the tag erases its identifier and does not answer until the next "first" request. If it is not the case, it follows the protocol. Note that the duration of the capacitor may be less than the duration of a singulation session if this capacity is reloaded periodically and the number of reloads is counted.

Threats Due to Lack of Randomness

Changing the tag identifier is essential but does not suffice because these identifiers need to be perfectly random not to supply an adversary with a source of additional information. The use of a cryptographically secure pseudo-random number generator (PRNG), initialized with a different value for every tag, is mandatory for avoiding traceability. Of course, singulation must rely only on this random identifier without requiring other characteristic data of the tag.

In the tree walking case, [53] proposes using short singulation identifiers that are refreshed for each new singulation using a PRNG. The used identifiers are short for efficiency reasons since there are usually only few tags in a given field. However, if the number of tags in the field is large, the reader can impose the use of additional static identifiers, available in the tag, set by the manufacturer! The benefit of using a PRNG is therefore totally null and void.

In the Aloha case, if the singulation identifiers do not appear in the acknowledgment sent by the readers, they do not directly bring information to an adversary. On the other hand, they supply much information through a side channel if we analyze how the slot is chosen by the tag. If the time slot is randomly picked, it will not supply useful information to the adversary, but a non uniform distribution can open the door to attacks. Unfortunately this is the case with current existing standards and protocols.

In order to illustrate our point, we can analyze the collision-avoidance protocol proposed by Philips for its tag ICode1 Label IC [144] using the 13.56 MHz frequency. It contains a 64 bit identifier of which only 32 are used for the singulation process, denoted by $b_1 ... b_{32}$. Although the tag does not use a PRNG for the singulation, the implemented collision-avoidance protocol is probabilistic. The choice of the time slot depends on the identifier of the tag and data sent by the reader. When the reader queries a tag, it sends a request containing: the number of slots n that the tags can use, where $n \in \{2^0, 2^1, ..., 2^8\}$, and a value $h \in 0, ..., 25$ called *hash value*. The selection of the time slot s_i is done as follows:

$$s_i := \text{CRC8}(b_{h+1} ... b_{h+8} \oplus \textit{prev}) \oplus n$$

where CRC8 is a *Cyclic Redundancy Check* with generator polynomial $x^8 + x^4 + x^3 + x^2 + 1$ and *prev* is the output of the previous CRC8, initialized with `0x01` when the tag enters the reader field. Hence, an adversary can easily track a tag according to the slot chosen by the tag, if she always sends the same values h and n. One way to proceed is as follows.

An adversary sends to her (isolated) targeted tag a request with the number of slots n and the hash value h. The tag responds during slot s_{target}. When she meets a set of m tags, the adversary wants to know if her target is in that set. In order to do this, she sends a singulation request containing the same n and h. If no tag responds during s_{target} then the target is not present in the set of tags. However, the conditional probability that the tag is in the set given that at least one tag answers during slot s_{target} is

$$P(n, m, p) = \frac{p}{p + (1-p)(1 - (\frac{n-1}{n})^m)},$$

where p is the probability that the target is in the set. Note that in the particular case of the ICode1 tag, where the CRC-8 is applied on a 8-bit word, we can actually recover 8 bits of the identifier by sending only one singulation request. Therefore, by sending 4 requests with respectively $h = 0$, $h = 8$, $h = 16$, and $h = 24$, the adversary will be able to recover the 32 bits of the tag's singulation identifier.

Imagine a scenario that illustrates this issue. In order to ensure the safety of an important personality during transportation, three cars that will take different routes will be used. Determining which vehicle will carry the personality will be done at the last moment. The adversary randomly chooses the car she will follow, but to avoid unnecessary risks, the adversary will attack the car only if she is almost sure that her man is really inside. The adversary will take advantage of the presence of a tag in the wristwatch of her potential victim. Beforehand, she has obtained, through a colluder who knows the target, the slot number s_{target} during which the tag responds when it is requested with $n = 256$ and $h = 0$. When she is close to the vehicle, the adversary queries the tags present in her environment. Suppose there are 10 tags in her field. If no tag responds during s_{target}, it means that her target is definitely not in the car. On the other hand, if at least one tag answers during s_{target}, then her target is in the car with probability $P(256, 10, 1/3) = 0.93$. This probability can be improved by querying the tags several times with different n and h values.

Consequently, choosing the identifier, in the case of the tree walking-based protocols, and choosing the time slot, in the case of the Aloha-based protocols, must be done using a cryptographically secure PRNG. Otherwise, an adversary may track her target in a probabilistic way, or worse, to recover its identifiers as with the ICode1 tag.

6.3.3 Traceability Within the Physical Layer

Most security threats can be solved independently from the physical layer, but when we speak of traceability, it is very different. The physical signals exchanged between a tag and a reader can allow an adversary to recognize a tag or a set of tags even if the information exchanged cannot be understood. All efforts to prevent traceability in the higher layers may be rendered useless if no care is taken at the physical layer.

Threats Due to Diversity of Standards

The radio transmission parameters (frequency, modulation, timings, etc.) follow given standards. Thus all tags using the same standard should send very similar signals. Signals from tags using different standards are easy to distinguish. A problem arises when we consider sets of tags rather than a single tag. In a few years, we may all be walking around with many tags in our belongings. If several standards are in use, each person may have a set of tags with a characteristic mix of standards. This mix of standards may allow a person to be traced. This method may be especially good at tracing certain types of people, like military forces or security personnel.

To reduce the threats of traceability due to characteristic groups of tags it is thus of paramount importance to reduce the diversity of the standards used in the market. Note that even if it is possible to agree on a single standard to use when RFID tags become popular, there will be times when a standard for a new generation of tags will be introduced. During the period of transition it will be possible to trace people due to characteristic mixes of old and new tags.

Threats Due to Radio Fingerprinting

Radio fingerprinting is a technique that has been used in mobile telephony to recognize cloned phones. By recording characteristic properties of the transmitted signals it is possible to tell a cloned cell-phone from the original one. Small differences in the transient behavior at the very beginning of a transmission allows for the identification of transceivers even if they are of the same brand and model [167]. In the case of RFID tags, there will be too many tags in circulation to make it possible to distinguish a single tag from all other tags of the same model. Nevertheless, there will be several manufacturers in the market and their tags will have different radio fingerprints. It will thus be possible to trace a person by a characteristic mix of tags from different manufacturers.

Preventing traceability through radio fingerprinting seems quite difficult. There is no benefit for the manufacturers to produce tags that use exactly the same technology, producing the same radio fingerprint. Much more likely, manufacturers will experiment with different technologies in order to produce tags that have either better performance, price or size.

Finally, even if avoiding traceability at the physical layer seems difficult to achieve, addressing this problem in the application and physical layers is important. Indeed, the difficulty to trace a tag increases as we travel downwards in the communication model's layers. Tracing a tag at the physical layer requires sophisticated material and in-depth knowledge, while tracing a tag at the application layer is much easier. By protecting privacy in the communication layer, more importantly in the application layer, we thus avoid attacks that any adversary could launch.

Protocols Based on Reader-Aided ID-Refreshment

CHAPTER SEVEN

To avoid a tag being traced, one manner is modifying its identifier (ID) such that only an authorized party is able to link the successive ID modifications.

A basic way is storing inside the tag a list of identifiers called pseudonyms, which are used sequentially and cyclically. This simple idea however requires a large amount of memory to store the pseudonyms, otherwise, the tag becomes traceable as soon as the pseudonyms have been used once.

A more sophisticated approach consists in refreshing the tag identifier using a deterministic or randomized process. This chapter studies six examples of protocols based on reader-aided ID-refreshment, that is the tags identifiers are refreshed by the reader, avoiding heavy computations on the tags. We present and analyze: Henrici and Müller's protocol [101] in Section 7.1; Golle, Jakobsson, Juels and Syverson's protocol [93] that relies on a *universal re-encryption* scheme, in Section 7.2; Saito, Ryou, and Sakurai's protocol [153] in Section 7.3; Juels's protocol [112] which uses only the XOR operation, in Section 7.4; Juels and Pappu's protocol [116] whose goal is to protect banknotes using RFID tags, in Section 7.5; and finally Yang, Park, Lee, Ren, and Kim 's protocol [111, 181] in Section 7.6. For each protocol, we exhibit weaknesses or attacks that endanger the privacy of the tags. In some cases, the threat appears because the authors focused on a too restricted adversary model that is not realistic. In some other cases the threat is due to weaknesses in the protocol design.

Whatever the protocol, they all suffer from a common weakness, inherent to the protocols that involve the reader in the refreshment process: the tag is always traceable between two legitimate identifications since the identifier is not refreshed during this period. As we have seen in Chapter 6, a legitimate identification is a successful identification between a tag and the system that owns or manages the tag. This problem is mitigated in [112] because the tag stores a small list of pseudonyms that gets renewed each time it is queried. However, as we will see below, this just pushes the problem back but does not solve it. Other attacks rely on the fact that the tag sends values that are distinguishable from random values, e.g., an identification session counter. Another important problem that we explain below is the desynchronization between the tag and the system: some protocols assume a kind of synchronization between the system and the tags it manages. For example, it can be the last used pseudonym or the number of the last successful identification. Finding a way to desynchronize the system and the tags is therefore a very efficient way to trace the tags because the identifiers can usually not be refreshed subsequently. Examples of these attacks are described below.

7.1 Henrici and Müller

7.1.1 Description

In Henrici and Müller's protocol [101], the tag needs to store a (non-static) identifier ID and two variables k and k_{last}. When the system is launched, the tag contains its current identifier ID, the current session number k (both are set up with random values), and k_{last} that is equal to k. On the reader's side, a database contains a similar 3-tuple per tag it manages. Data in the database and in the tags are initially synchronized. The tag identification works as follows (see Figure 7.1):

1. The reader sends a request to the tag.

2. The tag increases its current session number k by one and sends back $h(\text{ID})$, $h(k \oplus \text{ID})$ and $\Delta k := k - k_{\text{last}}$. $h(\text{ID})$ allows the database to recover the tag's identity; Δk allows the database to recover k and thus to compute $h(k \oplus \text{ID})$, whose goal is to thwart replay attacks.

3. The database checks the validity of these values according to its recorded data. If it matches, it sends a random number r and $h(r \oplus k \oplus \text{ID})$ to the tag and stores the new values[1]. Since the tag knows k and ID and receives r, it can check whether or not $h(r \oplus k \oplus \text{ID})$ is correct. If this is case, it replaces its identifier by $r \oplus \text{ID}$ and k_{last} by k. Otherwise it does not refresh its identifier.

7.1.2 Attack Based on Non-Random Information

This attack consists of tracking a tag, taking advantage of the information supplied by Δk. Indeed, the tag increases its value k every time it receives a request (Step 2) even when the identification fails, but it updates k_{last} only when the identification succeeds (Step 3). Thus, an adversary may query the tag several times to abnormally increase k and in turn Δk. Because this value is sent in clear in the second message, the adversary is then able to

[1] Note that due to resiliency considerations, the entry $(\text{ID}, k, k_{\text{last}})$ in the database is not erased when the database has replied to the tag, but a copy is kept until the next correct session: if the third step fails, the database will still be able to identify the tag the next time with the "old" entry. Thus two entries per tag are used in turn.

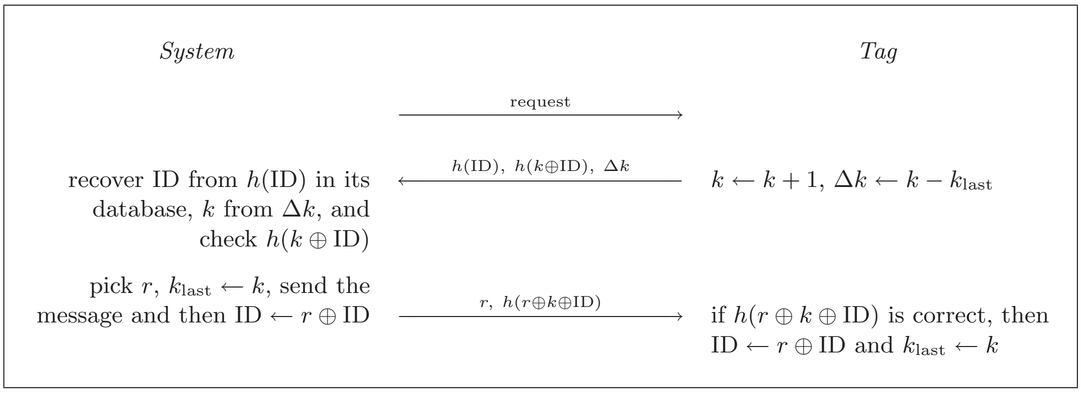

System
Tag
request
recover ID from $h(\mathrm{ID})$ in its database, k from Δk, and check $h(k \oplus \mathrm{ID})$
$h(\mathrm{ID}),\ h(k \oplus \mathrm{ID}),\ \Delta k$
$k \leftarrow k+1,\ \Delta k \leftarrow k - k_{\text{last}}$
pick r, $k_{\text{last}} \leftarrow k$, send the message and then $\mathrm{ID} \leftarrow r \oplus \mathrm{ID}$
$r,\ h(r \oplus k \oplus \mathrm{ID})$
if $h(r \oplus k \oplus \mathrm{ID})$ is correct, then $\mathrm{ID} \leftarrow r \oplus \mathrm{ID}$ and $k_{\text{last}} \leftarrow k$

Figure 7.1: Henrici and Müller's protocol

later recognize its target according to this value: if the tag sends an abnormally high Δi, the adversary concludes that this is her target, even in a passive way.

7.1.3 Attack Based on Refreshment Avoidance

Another attack consists of corrupting the hash value sent by the reader. When this value is not correct, "the message is discarded and no further action is taken" [101], so the tag does not refresh its identifier. Note however that it is easier to inject than to modify a message into a wireless channel. We therefore propose a practical variant of this attack: when a reader queries a tag, the adversary queries this tag as well before the reader carries out the third step. Receiving the request from the adversary, the tag increases k. Consequently, the hash value sent by the reader seems to be incorrect since k has now changed. More generally, an adversary can always trace a tag between two correct identifications. Combined with a relay attack, described in Chapter 6, this attack is even easier to put into practice.

7.1.4 Attack Based on Database Desynchronization

A more subtle and definitive attack consists of desynchronizing the tag and the database (as a kind of denial of service attack). For that, the adversary performs the identification so that the random value r she sends is the neutral element of $\oplus$: the adversary replaces r by the null bit-string and replaces $h(r \oplus k \oplus \mathrm{ID})$ by $h(k \oplus \mathrm{ID})$ obtained by eavesdropping the second message of the current identification. We have trivially $h(\mathbf{0} \oplus k \oplus \mathrm{ID}) = h(k \oplus \mathrm{ID})$. Hence, the tag cannot detect the attack. Then the tag replaces its identifier by $\mathbf{0} \oplus \mathrm{ID}$ (that is equal to its previous identifier) and it updates k_{last}. In the next identification, the tag and the database will be desynchronized since the tag computes the hash value using the previous ID and the fresh k_{last} whereas the database checks the hash value with the previous ID and the previous k_{last}: the test fails and the received message is discarded. Consequently, the database will never send the third message to refresh the tag's identifier and the tag is definitively traceable.

The above attack can be thwarted just by checking that $r \neq \mathbf{0}$. However, we show below that a desynchronization-based attack is still possible when $r \neq \mathbf{0}$. First of all, the adversary eavesdrops an interaction between her targeted tag and a legitimate reader. Let $h(k_i \oplus \mathrm{ID})$

and $\Delta k_i = k_i - k_{i-1}$ be the collected information. Later, the adversary queries the tag again, obtaining thus $h(k_j \oplus \mathrm{ID})$ and $\Delta k_j = k_j - k_{j-1}$. Given that

$$k_i - k_j = \sum_{\ell=i}^{j-1} \Delta k_\ell,$$

she guesses $k_i \oplus k_j$. For example, if $k_i - k_j = 1$ (which is the common case in close environments) then $k_i \oplus k_j = 00\ldots01$ with probability 1/2. While generating the third message, she takes $r = k_i \oplus k_j$ and $h(k_i \oplus \mathrm{ID})$. When it received the third message of the exchange, the tag checks whether the received hash value is valid, which is true since $h(r \oplus k \oplus \mathrm{ID}) = h(k_i \oplus k_j \oplus k_j \oplus \mathrm{ID}) = h(k_i \oplus \mathrm{ID})$. As in the case $r = \mathbf{0}$, the attack desynchronizes the database and the tag, which definitively becomes traceable.

7.2 Golle, Jakobsson, Juels and Syverson

7.2.1 Description

Golle *et al.*'s protocol [93] relies on the concept of *universal re-encryption*, i.e., a scheme where re-encryptions of a message m are performed neither requiring nor yielding knowledge of the public key under which m had been initially encrypted. The scheme consists of encrypting a plaintext m by appending two ciphertexts: the first one is the Elgamal encryption of m while the second one is the Elgamal encryption of the neutral element of $\mathcal{G}$, where $\mathcal{G}$ is the underlying group for the cryptosystem. We recall that encryption with the Elgamal scheme [68] of a message m under the public key y and a random number r is (my^r, g^r), where g is a generator of $\mathcal{G}$.

Below, we expand Golle *et al.*'s protocol. Let E be the Elgamal encryption scheme, and U be the corresponding re-encryption scheme, we have $U(m) := [E(m); E(1_{\mathcal{G}})]$. Let q be the order of $\mathcal{G}$, and g a generator. The universal re-encryption scheme is defined by the following four algorithms:

- *Key generation:* output the private key $x \in \mathbf{Z}$ and the public Elgamal key $y = g^x$.
- *Encryption:* let (r_0, r_1) be a random element picked in $(\mathbf{Z}/q\mathbf{Z})^2$. The encrypted value of a message m is

$$U(m) = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)] = [(my^{r_0}, g^{r_0}); (y^{r_1}, g^{r_1})].$$

- *Decryption:* given the ciphertext $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$, if α_0,β_0,α_1,$\beta_1 \in \mathcal{G}$ and $\alpha_1/\beta_1^x = 1$, then the plaintext is α_0/β_0^x.
- *Re-encryption:* let (r'_0, r'_1) be a random element picked in $(\mathbf{Z}/q\mathbf{Z})^2$. The re-encrypted value of a ciphertext $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ is

$$[(\alpha_0\alpha_1^{r'_0}, \beta_0\beta_1^{r'_0}); (\alpha_1^{r'_1}, \beta_1^{r'_1})].$$

We now describe the RFID protocol suggested by Golle *et al.*, based on their universal re-encryption scheme. During the tag initialization, an encrypted identifier is stored in the tag.

This encrypted identifier as well as the secret key corresponding to the tag are stored in the database. As depicted in Figure 7.2, an execution is carried out as follows: (1) The reader sends a request to the tag; (2) The tag sends back its encrypted identifier; (3) The reader re-encrypts the tag identifier using the universal re-encryption scheme described above and sends the new value to the tag (Figure 7.2).

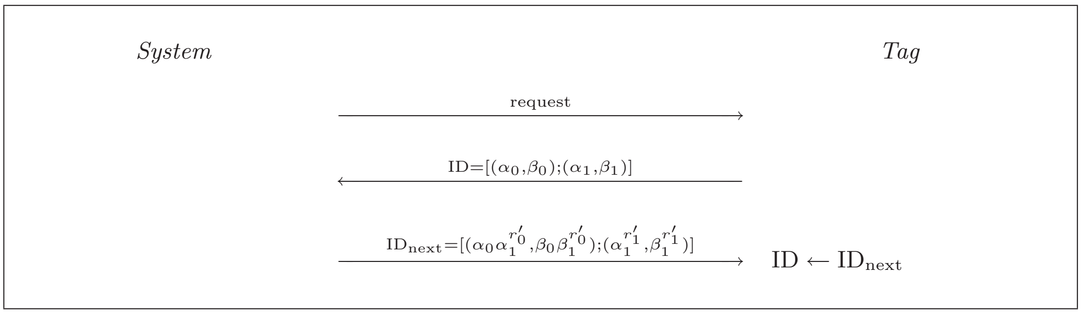

System
Tag
request
$\mathrm{ID}=[(\alpha_0,\beta_0);(\alpha_1,\beta_1)]$
$\mathrm{ID}_{\mathrm{next}}=[(\alpha_0\alpha_1^{r'_0},\beta_0\beta_1^{r'_0});(\alpha_1^{r'_1},\beta_1^{r'_1})]$
$\mathrm{ID} \leftarrow \mathrm{ID}_{\mathrm{next}}$

Figure 7.2: Golle, Jakobsson, Juels, and Syverson's protocol

As noticed in [93], if an adversary sends a fake re-encrypted identifier to the tag, the database will not be able to identify the tag in the future. The authors say that this attack does not allow the tag to be traced, it will only harm the normal functioning of the system. The authors do, however, reveal an exception: when an adversary replaces the value (α_1, β_1) by $(1_{\mathcal{G}}, 1_{\mathcal{G}})$ where $1_{\mathcal{G}}$ represents the neutral element of $\mathcal{G}$, the future re-encryptions will no longer change the identifier. The tag can protect itself from this attack by verifying that (α_1, β_1) is not equal to $(1_{\mathcal{G}}, 1_{\mathcal{G}})$ before changing its value. However, Golle *et al.*'s protocol also suffers from other weaknesses: Saito, Ryou, and Sakurai [153] stress that an adversary can replace the identifier of the tag by a value she has encrypted with her own public key. Thus, she is able afterwards to decrypt the content of the tag and trace it. Very recently, Ateniese, Camenisch, and de Medeiros [12] suggested an approach to thwart this attack. Their solution is based on a new cryptographic primitive, called *insubvertible encryption.*

We describe in Section 7.2.2 and Section 7.2.3 two other attacks against [93].

7.2.2 Attack Based on Eavesdropping

The first thing to notice is that the protocol [93] does not resist to simple eavesdropping attacks. Indeed, since the tag sends in the second message what it received in the third message of the previous execution, an adversary is able to trace the tag by eavesdropping the communication.

7.2.3 Attack Based on Invariants

The weakness described below results from the fact that the ciphertext sent by the tag is not random. Taken independently, every element of the ciphertext $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ follows a uniform distribution assuming that the discrete logarithm is a random function, but these elements are not independent.

We denote by $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$ the message sent by the tag during the i-th identification. If $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$ verifies a property $\mathcal{P}$ that is invariant by re-encryption, i.e.,

$\mathcal{P}$ remains verified after re-encryption, then the adversary is (almost certainly) able to trace the tag. We describe the attack below. Let us define $\mathcal{P}$ such as $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$ verifies $\mathcal{P}$ if and only if $\alpha_1 = \beta_1$. If $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$ verifies $\mathcal{P}$, then $[(\alpha_0^{(j)}, \beta_0^{(j)}); (\alpha_1^{(j)}, \beta_1^{(j)})]$ verifies it as well for any $j \geq i$. Indeed, the same (deterministic) operation is applied to both α_1 and β_1 during a re-encryption, that is both α_1 and β_1 are raised to a given power r'_1.

In order to trace a tag, the adversary queries it and sends the (third) message

$$\mathrm{ID}_{\mathrm{next}} = [(a, b); (c, c)]$$

where a, b, and c can be any values. When she queries the tag next and receives the message $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$, she verifies whether $\alpha_1^{(i)} = \beta_1^{(i)}$. In this case, the queried tag is her target with high probability. While the tag could detect such an attack by testing that $\mathrm{ID}_{\mathrm{next}}$ does not verify $\mathcal{P}$, there are other invariant properties, e.g., the property $\mathcal{P}'$ such that $[(\alpha_0^{(i)}, \beta_0^{(i)}); (\alpha_1^{(i)}, \beta_1^{(i)})]$ verifies $\mathcal{P}'$ if and only if $\alpha_1^{(i)} \cdot \beta_1^{(i)} = 1$ in $\mathcal{G}$.

7.3 Saito, Ryou, and Sakurai

As we have said in Section 7.2.1, Saito *et al.* also pointed out an attack against Golle *et al.*'s protocol. They subsequently suggested two RFID protocols based on [93]. The first one, described in Section 7.3.1, is called "with a check", and the second one, described in Section 7.3.2, is called "With One-Time Pad".

7.3.1 With a Check

The first protocol is an improvement of [93] where operations carried out by the tag have been modified: the tag checks the new value re-encrypted by the reader before accepting it as the new identifier. The aim is to detect an adversary who would send a wrong re-encrypted identifier. Therefore, when a tag is queried, it sends its current identifier, $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$, and receives the new value $[(\alpha'_0, \beta'_0); (\alpha'_1, \beta'_1)]$. If $|\alpha'_0|, |\beta'_0| \neq 1$ and if $\alpha'_0/{\beta'_0}^x = 1$, where x is the private key of the tag, then $[(\alpha'_0, \beta'_0); (\alpha'_1, \beta'_1)]$ becomes the new current identifier. If not, the tag does not renew its content.

7.3.2 With a One-Time Pad

The second protocol suggested by Saito *et al.* is also based on the universal re-encryption scheme introduced in [93]. The primary difference compared to [93] is that the re-encryptions are carried out by the tag itself and no longer by the reader. Since the tag is not able to carry out the exponentiations itself, pre-calculations are carried out by the database and sent to the tag from time to time. Below, we expand the protocol.

To begin with, the tag contains an identifier $\mathrm{ID} = [(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$. It also has a finite list of pairs of random values $\Delta = ((\alpha_1^{r_1}, \beta_1^{r_1}), (\alpha_1^{r_2}, \beta_1^{r_2}), \dots)$ that will allow it to re-encrypt its identifier. The tag also contains a variable k that is the session number, as well as a secret S. All these data are shared with the database. We must consider two distinct operations in this protocol: the reading of the tag and the update of its list of random values, which does not occur at every identification. The procedure unfolds in the following way (see Figure 7.3):

1. The reader sends a request to the tag.
2. The tag sends back ID and replaces its identifier by
$$\mathrm{ID}_{\mathrm{next}} := [(\alpha_0\alpha_1^{r_k}, \beta_0\beta_1^{r_k}); (\alpha_1\alpha_1^{r_{k+1}}, \beta_1\beta_1^{r_{k+1}})] \text{ where } (\alpha_1^{r_k}, \beta_1^{r_k}),\ (\alpha_1^{r_{k+1}}, \beta_1^{r_{k+1}}) \in \Delta.$$
3. If an update of Δ is needed, the reader sends a new list Δ_{next} of random values and the key $X = h(S, k, \Delta)$ to the tag, where h is a hash function. If the key is correct, then the tag replaces Δ by Δ_{next} and increments the session number k. If not, the tag does nothing.

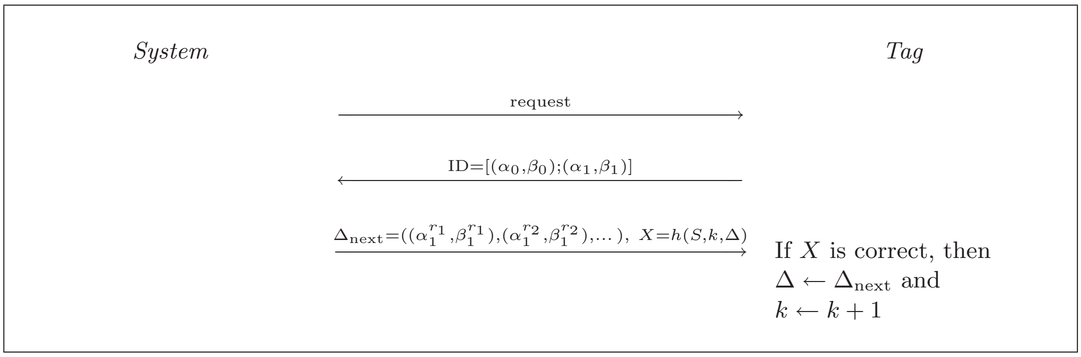

System
Tag
request
$\mathrm{ID}=[(\alpha_0,\beta_0);(\alpha_1,\beta_1)]$
$\Delta_{\mathrm{next}}=((\alpha_1^{r_1},\beta_1^{r_1}),(\alpha_1^{r_2},\beta_1^{r_2}),\dots),\ X=h(S,k,\Delta)$
If X is correct, then
$\Delta \leftarrow \Delta_{\mathrm{next}}$ and
$k \leftarrow k+1$

Figure 7.3: Saito, Ryou, and Sakurai's protocol

7.3.3 Attack Based on the Private Key

In Saito, Ryou, and Sakurai's first protocol [153], the fact that the tag carries out a test based on its public/private key transforms it into an oracle that responds whether or not this value has been encrypted with its public key. In other words, the oracle responds whether or not we are dealing with the traced tag. Let us however note that this response from the oracle is internal to the tag. The adversary therefore still has to recover this response. This is rather straightforward because the tag changes its identifier if and only if the test succeeds. So the adversary proceeds as follows. She requests its targeted tag for the first time thus obtaining a reference identifier $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$. Subsequently, when the adversary wants to know if a tag corresponds to her target, she queries it: she receives (message 2) a value $[(\alpha_0', \beta_0'); (\alpha_1', \beta_1')]$ and resends (message 3) the value $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ to the tag instead of resending the value $[(\alpha_0', \beta_0'); (\alpha_1', \beta_1')]$ re-encrypted. She queries the tag once again. If she again receives $[(\alpha_0', \beta_0'); (\alpha_1', \beta_1')]$, this means that the tag has not renewed its identifier and she is not dealing with the traced tag. The traced tag would have recognized $[(\alpha_0, \beta_0); (\alpha_1, \beta_1)]$ as a valid value, meaning encrypted with its public key, and would have used it to refresh its identifier.

7.3.4 Attack Based on the Random Values

In Saito, Ryou, and Sakurai's second protocol [153], knowing the list of the random values contained in the tag allows an adversary to easily trace a tag as she can calculate all the identifiers that will be used by it. So, eavesdropping the communication between the reader and the tag during an update is sufficient to subsequently trace the tag. Since the adversary has to be present during the update (which is only carried out from time to time), she can force the update using a man-in-the-middle attack. No authentication is used in the protocol. Thus the tag knows that Δ_{next} has been created by the database but it does not know who is sending it this value. On the other hand, the database does not know that it is sending Δ_{next} to the adversary instead of sending it to the tag. The session number prevents a replay-attack, not a man-in-the-middle attack.

7.3.5 Attack Based on Database Desynchronization

The danger that exists for protocols using synchronized values between the tag and the database (the session number k in Saito, Ryou, and Sakurai's second protocol [153]) is that an adversary can cause a desynchronization between the two parties. We have already pointed out such an attack against Henrici and Müller's protocol in Section 7.1.4. Here, if an adversary causes the database to send the update message while the tag cannot receive it, then the session number stored by the database will be higher than that stored by the tag. Consequently, all the subsequent updates will fail as the calculation of the key X, which authorizes the update, takes into account the current session number.

7.4 Juels' XOR-Based Protocol

In the protocol presented in Section 7.3.2, the tag identifier is refreshed from a list of random values. This list is updated from time to time thanks to the reader. This idea is also the approach taken by the XOR-based protocol of Juels [112] in order to refresh the tag identifier, which is then involved in a kind of challenge-response protocol.

7.4.1 Description

Roughly, the concept of [112] consists in storing inside the tag a list of pseudonyms $\alpha_1, ..., \alpha_k$. Each time the tag is queried by a reader, it uses a fresh pseudonym, cycling to the beginning of the list after k successive identifications. Because the tag's memory is strongly limited, only few pseudonyms can be stored. To overcome this issue, the pseudonyms are refreshed thanks to the reader. In order to avoid an adversary from refreshing herself the pseudonyms of the tag, a mutual tag–reader authentication is required. For that, each pseudonym α_i is associated with two random values β_i and γ_i, also stored inside the tag, which are also refreshed. Thus, after the setup phase, both tag and system contain k 3-tuples $(\alpha_i, \beta_i, \gamma_i)$. In order to refresh these values, a vector of m random values is associated with each of these ek values.

When the tag is queried during the $(i+1)$-th identification, it sends to the system $\alpha_{(i \bmod k)+1}$, as depicted in Figure 7.4 (in the figure, i is stored into the counter c, which is initially equal to 0). The system looks for $\alpha_{(i \bmod k)+1}$ in its database. If this entry exists, the system sends back $\beta_{(i \bmod k)+1}$. This value enables the tag to check the authenticity of the system. Then the tag sends $\gamma_{(i \bmod k)+1}$ to the system. This latter checks whether this value is the expected one and, in this case, sends $3k$ vectors of m fresh random values that are used to update the α_is, β_is, and γ_is. Let κ be such a value, $\Delta_\kappa = (\delta_\kappa^{(1)}, ...\delta_\kappa^{(m)})$ the vector associated to κ, and $\tilde{\Delta}_\kappa = (\tilde{\delta}_\kappa^{(1)}, ...\tilde{\delta}_\kappa^{(m)})$ the vector sent by the system to update Δ_κ. The update procedure works as follows:

1. $\delta_\kappa^i \leftarrow \delta_\kappa^{(i+1)} \oplus \tilde{\delta}_\kappa^{(i)}$ where $1 \leq i < m$,

2. $\delta_\kappa^m \leftarrow \tilde{\delta}_\kappa^{(m)}$, and

3. $\kappa \leftarrow \kappa \oplus \delta_\kappa^{(1)}$.

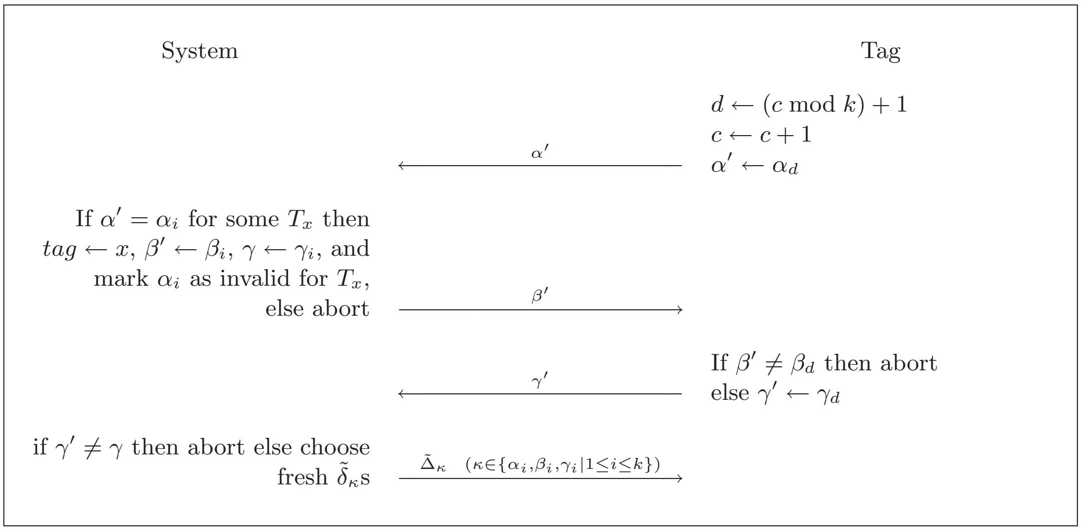

System
Tag
$d \leftarrow (c \bmod k) + 1$
$c \leftarrow c + 1$
$\alpha' \leftarrow \alpha_d$
α'
If $\alpha' = \alpha_i$ for some T_x then
$tag \leftarrow x$, $\beta' \leftarrow \beta_i$, $\gamma \leftarrow \gamma_i$, and
mark α_i as invalid for T_x,
else abort
β'
If $\beta' \neq \beta_d$ then abort
else $\gamma' \leftarrow \gamma_d$
γ'
if $\gamma' \neq \gamma$ then abort else choose
fresh $\tilde{\delta}_\kappa$s
$\tilde{\Delta}_\kappa \quad (\kappa \in \{\alpha_i, \beta_i, \gamma_i | 1 \leq i \leq k\})$

Figure 7.4: Juels' XOR-based protocol

7.4.2 Discussion

Since the number k of pseudonyms contained in the tag is small, an attack consists in discovering all the pseudonyms of the tag, which can be easily traced subsequently. Discovering all the pseudonyms can be done by querying at least k times the tag. This attack works when the tag is not queried by other readers during the attack or, more precisely, when the tag is not queried by legitimate readers. When a legitimate reader queries the tag, the pseudonyms are updated and the attack must be restarted from scratch. To make this attack harder, [112] suggests to impose a low query-response rate in the tag.

If the adversary is able to eavesdrop m successive interactions between the tag and a legitimate reader, then she is able to recover all the values contained in the tag. Consequently, she is able to impersonate the tag or to trace it. The threat remains as long as the adversary is able to eavesdrop all interactions between the tag and the legitimate readers. Note that interactions between the tag and foreign readers do not jeopardize the attack since updates are not performed in this case. As soon as the adversary misses one interaction between the tag and a legitimate reader, she must restart her attack, saying she must eavesdrop again m interactions. However, since the adversary is able to recover all the values contained in the tag after having observed m successive interactions, she is also able to herself update the tag's content. Subsequently, system and tag are desynchronized, meaning that the system is no longer able to find the pseudonyms of the tag in its database. Consequently, the pseudonyms of the tag are no longer refreshed, and it becomes traceable definitively.

Let's finally point out that this scheme is rather inefficient. Each identification requires exchanging $3km$ values, which must be stored by the tag. A few improvements are suggested in [112] but the communication complexity remains linear in k and m. Thus, the adversary can totally smash the purpose of this scheme with a number of interactions tag–legitimate reader equal to the number of 3-tuples stored in the tag. [112] defends itself arguing that eavesdropping successive interactions is a difficult task in practice due to the mobility of the tag, e.g., in a building access control system and due to the fact that the adversary must be between the tag and the legitimate reader. However, this difficulty can be easily overcome by using a relay attack, as discussed in Chapter 6.

Thus, this protocol is only resistant in a weak adversary model: limited successive tag queries and limited successive eavesdropped interactions between the tag and legitimate readers.

7.5 Juels and Pappu's Banknote Protection Scheme

At Financial Cryptography 2003, Juels and Pappu proposed a practical cryptographic banknote protection scheme [116] based on both optical and radio frequency identification systems. This scheme has been published after a rumor that appeared in 2001 saying that the European Central Bank (ECB) decided to use RFID tags to protect Euro banknotes.

However, as we explain in [15], this protocol severely compromises the privacy of the banknotes' bearers. We describe in this section some threats and show that, due to the misuse of Fujisaki and Okamoto's secure integration method, an adversary can access and modify the data stored in the smart device without optical access to the banknote. We also prove that despite what the authors claim, an adversary can track the banknotes by using the access-key as a marker, circumventing the randomized encryption scheme that aims at thwarting such attacks. Below, we introduce several attacks that can be performed on the Juels – Pappu banknote protection scheme. While some of these attacks are proper to that scheme (Section 7.5.5, and Section 7.5.6), some others are more general and could be applied to other RFID-based privacy protection schemes (Section 7.5.4, Section 7.5.7, Section 7.5.8, and Section 7.5.9).

7.5.1 Overview

Involved Parties

The goal of the protocol given in [116] is to resist banknotes counterfeiting and track traffic flows by some law enforcement agency, nevertheless guaranteeing the privacy of the banknote handlers. First we describe all the interested parties who are involved in the scheme.

- *Central bank.* The central bank aims at creating banknotes and at avoiding banknote forgery. It is therefore in its interest to have unique banknote serial numbers and to protect the secret key, which is used to sign the banknotes.

- *Law enforcement agency.* The goal of this agency is to arrest forgers. In order to achieve this, it needs to track banknotes and detect fake ones easily, even in areas of dense traffic, such as airports.
- *Merchants.* Merchants handle large quantities of banknotes. It is conceivable that they will try to compromise their clients' privacy. Merchants may comply with the law enforcement agency by reporting irregularities in banknote data.
- *Consumers.* Banknote bearers want to protect their privacy. Therefore, they want to limit banknotes tracking even if it means not respecting existing laws.

Concept and Requirements

Up to now, banknote security solely relies on optical features, which can be checked either by human-scanning or machine-scanning. In [116], security relies on both optical and electronic features. Banknotes thus have two data sources:

- *Optical*: data can be encoded in a human-readable form and/or in a machine-readable form such as a two-dimensional barcode. It contains banknote serial number as well as denomination, origin, etc.
- *Electronic*: data can be read by wireless communication. Data are signed by the central bank and encrypted with the law enforcement agency's public key and a random number.

Electronic data are stored in an RFID tag, which consists here of two cells whose access is key-protected. The access-key can be (re-)generated from the banknote optical data. One of the two cells, denoted γ, is universally readable but keyed-writable. The other cell, denoted δ, is both keyed-readable and keyed-writable. In [116], the proposed scheme consists in writing in γ the banknote's serial number signed by the central bank and encrypted with the law enforcement agency's public key. If this encrypted value was static, then an adversary could still track the banknote using this value as a marker. To overcome this weakness, the signature on the serial number is re-encrypted by merchants as often as possible, using obviously a probabilistic encryption scheme. Since the signature is available from the optical data, encryption is performed from scratch and does not need to be homomorphic. After the re-encryption is performed, the new encrypted value is put into γ and the used random value r is put into δ. Since γ and δ are keyed-writable, to re-encrypt the banknote, one must have optical contact with it to obtain the access-key. We will detail this procedure in Section 7.5.2.

Below, we give the requirements that [116] should guarantee:

- *Consumer privacy.* Only the law enforcement agency is able to track the banknotes using the RFID interface. Even the central bank is not allowed to track banknotes.
- *Strong tracking.* The law enforcement agency is able to identify a banknote (by its serial number) even without optical contact.
- *Minimal infrastructure.* In order to be user-friendly, the system should not require that banknote bearers possess special equipment. For their part, retail banks and shops should only buy devices at reasonable cost. Furthermore, they should not be required to set up a permanent network connection.

- *Forgery resistance.* A forger has to have optical contact with a banknote in order to create a fake one with the same serial number. A forger should not be able to create a fake banknote with a new serial number and moreover, she should not be able to change the banknote denomination.
- *Privilege separation.* The data stored in the tag should only be alterable given optical contact with banknotes.
- *Fraud detection.* If the data stored by the tag is wrong, then a merchant who has optical access to the banknote should be able to detect the forgery.

In order to illustrate these requirements, Juels and Pappu give two examples of privacy attacks that a banknote protection system should withstand. We recall these two examples here because we will show, in Section 7.5, that their scheme is actually not resistant to these attacks.

Example 6. *"Bar X wishes to sell information about its patrons to local Merchant Y. The bar requires patrons to have their drivers' licenses scanned before they are admitted (ostensibly to verify that they are of legal drinking age). At this time, their names, addresses, and dates of birth are recorded. At the same time, Bar X scans the serial numbers of the RFID tags of banknotes carried by its patrons, thereby establishing a link between identities and serial numbers. Merchant Y similarly records banknote serial numbers of customers from RFID tags. Bar X sells to Merchant Y the address and birth-date data it has collected over the past few days (over which period of time banknotes are likely not yet to have changed hands). In cases where Bar X and Merchant Y hold common serial numbers, Merchant Y can send mailings directly to customers indeed, even to those customers who merely enter or pass by Merchant Y's shops without buying anything. Merchant Y can even tailor mailings according to the ages of targeted customers. Patrons of Bar X and Merchant Y might be entirely unaware of the information harvesting described in this example."*

Example 7. *"A private detective wishes to know whether Bob is conducting large-value cash transactions at Carl's store. She surreptitiously intercepts the serial numbers on banknotes withdrawn by Bob and also records the serial numbers of those brought by Carl out of his store. If there is any overlap between sets of numbers, she concludes that Bob has given money to Carl. The private detective might reach the same conclusion if Bob leaves without banknotes that he carried into Carl's store. The private detective might also try to reduce her risk of detection by reading the banknotes of Bob and Carl at separate times, e.g., en route to or from the bank."*

Interface

Below, we give the common commands that are available on a tag:

- `read`: allows every reader to obtain the data stored in the tag memory.
- `write`: allows every reader to write data in the tag memory.

Some other commands can be available on a tag:

- `sleep`: this command is keyed so that the reader has to send a key in order to put the tag into the sleep state. Then the tag does not respond to the reader's queries until it receives the `wake` command with the legitimate key.
- `wake`: after this command, the tag starts afresh to respond to the reader. It is a keyed command associated with the `sleep` command.
- `kill`: this command destroys the tag definitively.

Moreover, Juels and Pappu [116] suppose that the following commands are available:

- `keyed-read`
- `keyed-write`

These commands are similar to the `read`/`write` commands, except that they are keyed.

7.5.2 Description of the Method

We explain in this section the operations that should be performed on the banknote. Let $\mathsf{Sign}(k, m)$ be the signature on a message m with a key k and $\mathsf{Enc}(k, m, r)$ the encryption of m under the key k with the random number r. We note $||$ the concatenation of two bit-strings.

Setup

Central bank $\mathcal{B}$ and law enforcement agency $\mathcal{L}$ own a pair of public/private keys $(PK_{\mathcal{B}}, SK_{\mathcal{B}})$ and $(PK_{\mathcal{L}}, SK_{\mathcal{L}})$ respectively. $PK_{\mathcal{B}}$ and $PK_{\mathcal{L}}$ are published as well as a collision-resistant hash function h.

Banknote Creation

For every banknote i, $\mathcal{B}$ selects (according to its own rules – which can be assumed to be public) a unique serial number S_i and computes its signature $\Sigma_i = \mathsf{Sign}(SK_{\mathcal{B}}, S_i||den_i)$ where den_i is the banknote denomination. $\mathcal{B}$ then computes an access-key D_i such that $D_i = h(\Sigma_i)^2$, prints S_i and Σ_i on the banknote, and computes

$$C_i = \mathsf{Enc}(PK_{\mathcal{L}}, \Sigma_i||S_i, r_i)$$

where r_i is a random number. C_i is written into γ and r_i is written into δ. Note that the access-keys D_i is not stored in the databases of $\mathcal{B}$. In order to keep in mind the values stored on/in the banknote, we give in Figure 7.5, established from [116], the content of the optical information as well as those of cells γ and δ.

[2] Juels and Pappu point out that it is important that the hash function be applied on Σ_i rather than on S_i because an adversary who knows a serial number would be able to compute the corresponding access-key without any optical contact with the banknote.

RFID	
Cell γ *universally-readable / keyed-writable*	Cell δ *keyed-readable / keyed-writable*
$C = \mathsf{Enc}(PK_{\mathcal{L}}, \Sigma \| S, r)$	r

Optical	
S	$\Sigma = \mathsf{Sign}(SK_{\mathcal{B}}, S \| den)$

Figure 7.5: Optical and RFID data

Banknote Verification and Anonymization

When a merchant $\mathcal{M}$ receives a banknote, he verifies it and then re-encrypts it according to the following steps:

1. $\mathcal{M}$ reads the optical data S_i and Σ_i and computes $D_i = h(\Sigma_i)$.

2. $\mathcal{M}$ reads C_i, stored in γ, and keyed-reads r_i which is stored in δ.

3. $\mathcal{M}$ checks that $C_i = \mathsf{Enc}(PK_{\mathcal{L}}, \Sigma_i \| S_i, r_i)$.

4. $\mathcal{M}$ randomly chooses r'_i and keyed-writes it into δ.

5. $\mathcal{M}$ computes $C'_i = \mathsf{Enc}(PK_{\mathcal{L}}, \Sigma_i \| S_i, r'_i)$ and keyed-writes it into γ.

If one of these steps fails then the merchant should warn the law enforcement agency.

Banknote Tracking

Let us consider a target banknote that the law enforcement agency $\mathcal{L}$ wants to check or track. $\mathcal{L}$ is able to easily obtain the cipher C reading the cell γ and then computes the plaintext $\Sigma \| S = \mathsf{Dec}(SK_{\mathcal{L}}, C)$. $\mathcal{L}$ can then check whether or not Σ is a valid signature. If Σ is valid then $\mathcal{L}$ obtains the banknote serial number S.

7.5.3 Cryptographic Algorithms

Encryption and signature schemes can be chosen among existing secure schemes. However they should bring security without involving high overhead. Juels and Pappu suggest using an Elgamal-based encryption scheme [68] and the Boneh–Shacham–Lynn signature scheme [45], both using elliptic curves. Let $\mathcal{G}$ denote an elliptic-curve-based group with prime order q and let P be a generator of $\mathcal{G}$. Let $SK_{\mathcal{L}} = x \in_R \mathbf{Z}/q\mathbf{Z}$ be the law enforcement agency's private key and $PK_{\mathcal{L}} = Y = xP$ the corresponding public key. A message $m \in \{0,1\}^n$ where n is reasonably sized, is encrypted with the Elgamal scheme under the random number r as follows:

$$\mathsf{Enc}(PK_{\mathcal{L}}, m, r) = (m + rY, rP).$$

Since Elgamal encryption scheme is not secure against adaptive chosen-ciphertext attacks, Juels and Pappu suggest using Fujisaki and Okamoto's secure integration method [84]. The message m is then encrypted as follows:

$$\mathsf{Enc}^*(PK_{\mathcal{L}}, m, r) = (\mathsf{Enc}(PK_{\mathcal{L}}, r, h_1(r||m)), h_2(r) \oplus m)$$

where h_1 and h_2 are two hash functions from $\{0,1\}^*$ to $\{0,1\}^n$. As explained in [116], signature size could be 154 bits. Assuming that a serial number can be encoded over 40 bits, the plaintext $\Sigma||S$ requires 194 bits. Let us consider a 195 bit order elliptic curve group. The size of $\mathsf{Enc}^*(PK_{\mathcal{L}}, \Sigma||S, r)$ will be 585 bits. The total required size will thus be 780 bits (585 bits in γ and 195 bits in δ). As pointed out in [116], current RFID tags can provide such resources, requiring an additional cost.

7.5.4 Pickpocketing Attack

This attack that Juels and Pappu already mentioned is significant enough to be recalled here. It requires an adversary to test a passer-by in order to detect if he carries some banknotes. Even if the adversary is not able to discover neither the serial number nor the denomination, she is able to establish how many banknotes the passer-by is bearing.

The adversary has less information if banknotes of all denominations are tagged than if only the largest ones are tagged. However, tagging banknotes of all denominations may be dangerous with the Juels – Pappu scheme due to the fact that scanning banknotes takes some time. Merchants would not agree to re-encrypt notes of small denominations; privacy could consequently be threatened.

Example 8. *Some criminals want to steal some cars in a car park. During daylight hours, they only break into a few cars so as not to attract attention. Their problem is therefore to determine which cars could be the "best" ones, that is the cars that contain currency. Thanks to the banknote protection system, they can radio-scan numerous cars in order to pinpoint their targets.*

These "pickpocketing" attacks show that the attack described in the second example of Juels and Pappu (page 96) still occurs even using their banknote protection scheme.

7.5.5 Data Recovery Attack

The data recovery attack consists of two steps: the first one aims at obtaining the access-key D and then the random number r that is stored in the δ-cell; the second step exploits a misuse of Fujisaki and Okamoto's secure integration method, in order to recover S and Σ. So, the adversary can obtain the serial number of the banknote without optical access to it. Note that even well-behaving merchants are not supposed to obtain this information from the electronic data.

Step 1

One of the goals of the scheme is to avoid γ-write-access without optical reading of the banknote. This implies that an adversary must have physical access to the banknote to modify the γ-cell. However a merchant who is willing to re-encrypt the banknote sends the access-key $D = h(\Sigma)$ (obtained by optical reading) to the tag in order to receive the value stored in the δ-cell, i.e., the random number r. The adversary can just eavesdrop the communication in order to steal D and then she is able to communicate with the tag and finally obtain the δ-cell value r. To buttress our argument, remember that it is usually easier to eavesdrop the forward channel, that is from the reader to the tag, than the backward channel. Note also that the communication range should be long enough to enable the law enforcement agency to track the banknotes even in areas of dense traffic, such as airports.

Step 2

The adversary first obtains the value stored in the γ-cell (universally readable). γ-cell contains:

$$\begin{aligned}\mathsf{Enc}^*(PK_{\mathcal{L}}, m, r) &= (\mathsf{Enc}(PK_{\mathcal{L}}, r, h_1(r||m)), h_2(r) \oplus m)\\ &= (r + h_1(r||m)PK_{\mathcal{L}},\ h_1(r||m)P,\ h_2(r) \oplus m)\end{aligned}$$

Notation is defined in Section 7.5.3. Let us consider $(\epsilon_1, \epsilon_2, \epsilon_3)$ such that

$$(\epsilon_1, \epsilon_2, \epsilon_3) = \mathsf{Enc}^*(PK_{\mathcal{L}}, m, r).$$

So:

$$\epsilon_1 = r + h_1(r||m)PK_{\mathcal{L}},\ \epsilon_2 = h_1(r||m)P, \text{ and } \epsilon_3 = h_2(r) \oplus m.$$

She obtains therefore

$$m = \epsilon_3 \oplus h_2(r) \text{ where } \epsilon_3, r, \text{ and } h_2 \text{ are known.}$$

Since $m := \Sigma||S$, this proves that an adversary can discover the serial number and the signature of a banknote without having optical access to it, contrary to what Juels and Pappu claim.

The problem arises from the fact that Fujisaki and Okamoto's integration method is not secure anymore when the random value r is revealed. Indeed, the purpose of the asymmetric encryption $\mathsf{Enc}(PK_{\mathcal{L}}, r, h_1(r||m))$ is to "hide" the random value that is used to generate the key of the symmetric encryption. If this random value is public or can be easily determined by an adversary, the integration method becomes null and void.

7.5.6 Ciphertext Tracking

The protection method that we discuss here uses re-encryptions to prevent tracking attacks. However, re-encryptions can only be performed in practice by merchants or by retail banks[3]. Therefore the time period between two re-encryptions could be long enough to track banknotes.

[3] We could imagine a scenario where citizens are able to re-encrypt their own banknotes, but it is an unrealistic assumption.

Example 9. *Many supermarkets use massive computing power to analyze the buying patterns of their clients. Identifying these patterns enables merchants to reorganize their store layouts to increase their sales. Data mining consists of using computer-based search techniques to sort through the mounds of transaction data captured through goods bar-coding. The frequently cited example is the "beer and diapers" example: a large discount chain discovered by data mining its sales data that there was a correlation between beer and diaper purchases during the evening hours. The discount chain therefore moved the beer next to the diapers and increased sales. Let us now consider a merchant who installs RFID readers in his store departments: now he is able to analyze precisely his client's path and thereby to reorganize his store layout. Since some existing payment systems contain names and addresses, a client who stays during a long time in the bicycle department without buying anything will receive directly advertising literature to his door.*

Ciphertext tracking attacks show that the threat described in the first example of Juels and Pappu (page 96) still occurs within their banknote protection scheme. Let us first consider a milder version of the attack: bar X cannot read the optical data on the banknotes of his customers (we consider that a *customer* is a person who comes in the shop; he does not necessarily need to buy anything). So, he stores in a database all the γ-values that he is able to collect matched with the name and address of their handlers. Merchant Y also reads the γ-values of his clients and stores them. Bar X and merchant Y can combine their databases: if a γ-value appears in both databases, they are almost sure that it is the same client. Let us now consider a stronger attack: when bar X returns change to a client, he re-encrypts banknotes with a fixed number, denoted r_0 also known by merchant Y. When a customer arrives in Merchant Y's store, the merchant reads the γ-value of the customer's banknotes (universally readable) and computes Σ_0 using r_0 (applying the method described in Section 7.5.5). He then computes $D_0 = h(\Sigma_0)$ and tries to read δ with D_0; if the tag agrees this means that r_0 was the appropriate random number and that merchant Y can be almost sure that this client comes from Bar X. Note that Merchant does not "touch" the banknote here: he has just to scan the people when they pass through the store door for instance.

This issue is inherent in reader-aided ID-refreshment schemes: since re-encryptions cannot be performed very frequently, it is possible to track tags with their universally readable values (even if these values seem to be some garbage for a person who is not authorized to decrypt them). Note that even with a higher re-encryption frequency, the attack still works if the re-encryptions are performed by the merchants, and not by the users themselves.

7.5.7 Access-Key Tracking

The goal of the re-encryptions is to prevent banknotes tracking, as we mentioned. If an adversary does not have optical contact with a given banknote, then she should not be able to track it in the long-term. Unfortunately, we demonstrate here that a side channel can be used to track the banknotes. Indeed, if the adversary can see the banknote once (or even, more simply, if she effects the first step of the attack described in Section 7.5.5) then thanks to the static access-key D, she will be able to track the banknote by just trying to read the δ-cell: the tag responds if and only if the key D is the good one.

This attack is particularly devastating because it dashes the purpose of the scheme. Actually, when a tag owns a unique access-key and responds if and only if the key sent by the reader is the valid one, this key can be used to track the tag. A solution could be to reply with some garbage when the access-key is wrong, instead of remaining silent.

Example 10. *Mrs Johnson suspects that her husband is having an affair with his secretary. It seems that he has been giving her money. Mrs Johnson decides to read the optical data on her husband's banknotes - in order to generate the access-key - and to surreptitiously follow his secretary after work. She will soon know whether her suspicions are true or not.*

7.5.8 Cookies Threat

According to [116], the sizes of the δ-cell and γ-cell are 195 bits and 585 bits respectively. Since these values can be modified for everyone having access to the banknote (or using the attack described in Section 7.5.5), the δ-cell and the γ-cell can be used to hide a certain amount of information. This hidden information channel looks like an HTTP cookie. This cookie will however be detected during the next re-encryption of the tag data (since merchants have to check the current value before performing the re-encryption) because δ and γ are not consistent anymore.

A clever way to abuse the tag is to put the cookie only in the δ-cell: since the value r stored in this cell is a random number, it can be used to store some information. Obviously, the γ-cell value will have to be re-encrypted with the new random number. This kind of cookie will be untraceable and will stay available until the next re-encryption.

7.5.9 Denial of Service Attack

We saw that when a merchant finds a discrepancy on a banknote, he cannot accept the payment and should warn the law enforcement agency. This could however be used to harm banknote bearers: all that is required is to input incorrect data into either δ or γ. This could be done not only by a merchant who has access to the optical data but also by anyone who is able to perform the first step of the attack described in Section 7.5.5. Due to its simplicity, this malicious attack may bring about many problems as the law enforcement agency as well as the Central Bank – that has to restore the banknotes – would be flooded.

Example 11. *In a store, some hackers are waiting in line for the cash register. The customer in front of them pays for his purchases. The hackers eavesdrop the communication between the reader and the tag, thus obtaining the access-key D; they then replace the data stored in the cell γ with a false value, just for fun. The banknote becomes out of service until it is restored by the Central Bank. They can block all cashiers this way and take advantage of panic scenes between cashiers and complaining customers in order to steal goods.*

7.5.10 Discussion

Two parties can benefit directly from the use of tags in banknotes: the central bank and the law enforcement agency, both profiting from this system by enforcing banknote tracking and anti-counterfeiting. What about the other interested parties such as merchants and citizens?

The role of merchants here is crucial since privacy relies only on their collaboration. Even if most merchants are compliant, one can notice that re-encrypting banknotes implies a loss of time for merchants. Another issue is the attitude of a merchant when faced with a problematic banknote; according to [116], he should warn the law enforcement agency. However he is able to repair the problematic data as he has optical access to the banknote. Will he risk losing a customer by warning the law enforcement agency?

From the citizens point of view, it would be difficult to tolerate such a system for three reasons. The first one is that citizens have to travel to the central bank (or perhaps to a retail bank) every time they have a faulty banknote. The second reason is that they will lose confidence in their banknotes: they can no longer be sure that they will be able to use their currency at the cash register! Last but not least, they will be suspicious about the fact that their privacy and their anonymity remain intact.

Finally, we underline that Zhang and King [185] have recently proposed an improvement to [116], which avoids an adversary swapping values between two banknotes without being detected.

7.6 Yang, Park, Lee, Ren, and Kim

7.6.1 Description

In order to conclude this chapter devoted to RFID protocols where the tag is aided by the reader to refresh its identifier, we put forward a rather straightforward attack against Yang *et al.* [111, 181]'s protocol.

As in all the previously presented protocols, an adversary can trace the tag between two legitimate identifications. In other words, if the adversary is able to *query* the tag at time t_1, then she is able to trace the tag at time $t_2 \neq t_1$ if and only if there is no legitimate identification between t_1 and t_2. We give here the following variant: if the adversary is able to *eavesdrop* a legitimate identification at time t_1, then she is able to trace the tag at time $t_2 > t_1$ if and only if there is no legitimate identification between t_1 and t_2. Below, we explain our attack.

The authors of [111, 181] distinguish the reader from the back-end database whereas up until now, we have considered them as a unique entity. In Figure 7.6, we only illustrate the exchanges between the reader and the tag because the adversary does not need to exploit the channel between the reader and the back-end database in order to carry out her attack. At the beginning, the tag and the system share 3 random values k_1, k_2, and C. k_1 and k_2 are refreshed during each legitimate identification while C is a static value. Three exchanges are required in this protocol:

1. The system queries the tag with a value S. The precise content of S is not relevant in the attack. More details are available in [111, 181].

2. The tag answers with $\mathrm{ID} = h(k_1 \oplus S \oplus C)$ where h is a hash function. If the message is the expected one, the system computes $\mathrm{ID}' := h(k_2)$.

3. Finally, the system sends ID' to the tag and the latter replaces k_1 by $k_1 \oplus \mathrm{ID}'$ and k_2 by $k_2 \oplus \mathrm{ID}$ if the received ID' is valid.

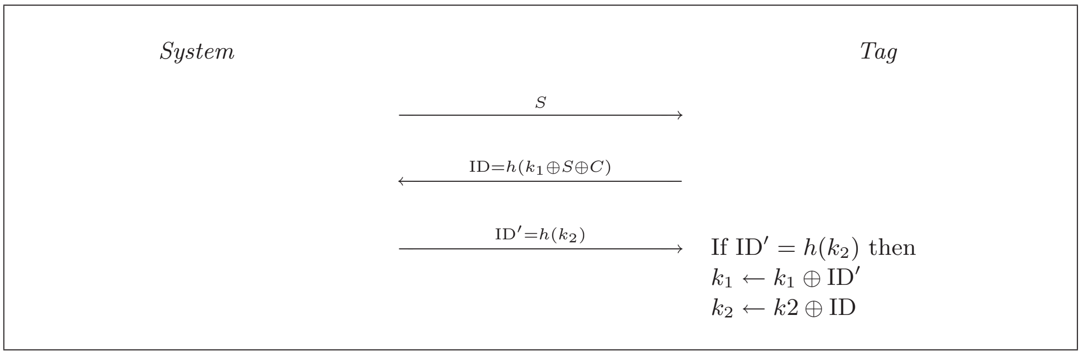

System
Tag
S
$\mathrm{ID}{=}h(k_1{\oplus}S{\oplus}C)$
$\mathrm{ID}'{=}h(k_2)$
If $\mathrm{ID}' = h(k_2)$ then
$k_1 \leftarrow k_1 \oplus \mathrm{ID}'$
$k_2 \leftarrow k2 \oplus \mathrm{ID}$

Figure 7.6: Yang, Park, Lee, Ren, and Kim's protocol

7.6.2 Attack

Our very simple attack works as follows. If the adversary is able to eavesdrop a legitimate identification, she obtains the current exchanged values S_{cur}, ID_{cur}, and $\mathrm{ID}'_{\text{cur}}$. Later, she queries the tag with $S_{\text{next}} = S_{\text{cur}} \oplus \mathrm{ID}'_{\text{cur}}$. Subsequently, the tag computes $\mathrm{ID}_{\text{next}} := h((k_1 \oplus \mathrm{ID}'_{\text{cur}}) \oplus (S_{\text{cur}} \oplus \mathrm{ID}'_{\text{cur}}) \oplus C)$, which is clearly equal to ID_{cur}. Thus, if $\mathrm{ID}_{\text{cur}} = \mathrm{ID}_{\text{next}}$, the adversary is certain that it is the same tag in both identifications.

Protocols Based on Self-Refreshment and Complexity Issues

CHAPTER EIGHT

In this chapter, we present radio frequency identification protocols based on self-refreshment, that is, protocols in which the tag refreshes its identifier by itself, without the help of the reader. These protocols rely on a challenge-response exchange and assure either identification or authentication, as defined in Chapter 6. Such an approach avoids the attacks presented in Chapter 7 against protocols based on reader-aided ID-refreshment. Proof of security can even be supplied under certain assumptions. Thus, protocols based on self-refreshment seem to be precisely what we need both in theory and in practice. However, below we show that these protocols require a linear computation complexity on the system's side in order to identify only a single tag. Moreover, they require computations on the tag's side that are usually based on a pseudo-random function. Such a function is still rather heavy to implement in very low cost tags [2, 65, 73] and can only deal with tags that are a bit more expensive.

Below, we present RFID (identification) protocols that require 2 moves [138, 178] and RFID (authentication) protocols that require at least 3 moves [73, 118, 136, 151]. For each of them, we describe the protocol by defining three phases: (a) the Setup phase where the database of the system and the tags are initialized; (b) the Interaction phase where the system and the tag interact; and (c) the Search phase where the system looks for the tag identity in its database. Sometimes, Interaction and Search phases are interleaved. Finally, we set out the complexity issue, which will be addressed in Chapter 9.

8.1 Weis, Sarma, Rivest, and Engels

In this section, we describe Weis, Sarma, Rivest, and Engels's protocol [178] with "Randomized Access Control". In this protocol (see Figure 8.1), the information sent by the tag each time it is queried consists of a random value a and a randomized hash value $\sigma = h(\mathrm{ID}||a)$ where ID is the static identifier of the tag. In order to compute this information, the tag needs a pseudo-random number generator and an embedded one-way hash function but only stores its identifier. Below we give the three phases of the protocol.

Setup

Each tag is initialized with a randomly chosen identifier ID. For each tag it manages, the system stores an entry in its database that contains its identifier.

Interaction

The system sends a request to the tag. Upon reception of this message, the tag picks a random a and computes $\sigma = h(\mathrm{ID}||a)$. It sends both a and σ to the system.

Search

Upon reception of σ and a, the system performs an exhaustive search in its database: for each entry ID, it computes $h(\mathrm{ID}||a)$ until it finds σ.

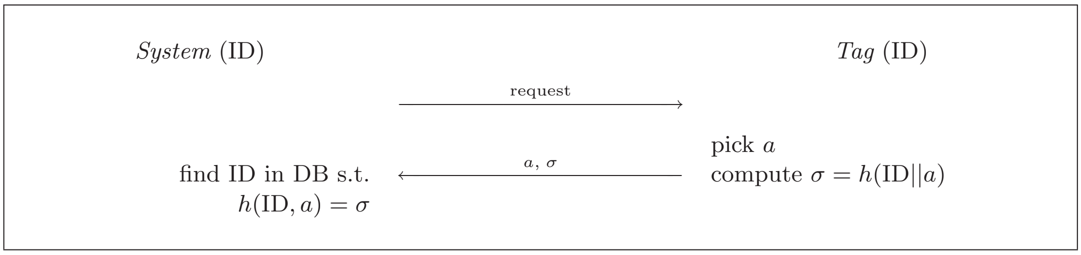


Figure 8.1: Weis, Sarma, Rivest, and Engels's protocol using a hash function

Weis *et al.* emphasize that from a theoretical point of view, the hash functions guarantee by definition irreversibility, but not secrecy: input bits may be revealed. Consequently, they suggest another construction that relies on pseudo-random functions. In that variant, the tag shares a secret s with the database and, instead of sending a and $h(\mathrm{ID}||a)$, the tag sends a and $\mathrm{ID} \oplus f_s(a)$ where f_s is a pseudo-random function chosen in a set $\mathcal{F} = \{f_s\}_{s \in \mathbf{N}}$. The exchanges are depicted in Figure 8.2

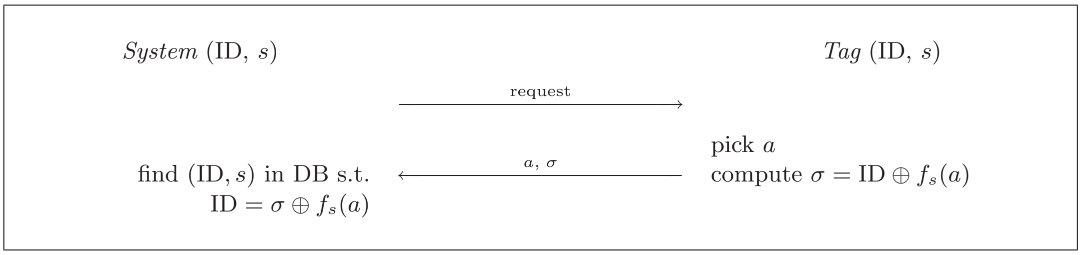

System (ID, s)
Tag (ID, s)
request
pick a
compute $\sigma = \mathrm{ID} \oplus f_s(a)$
a, σ
find (ID, s) in DB s.t.
$\mathrm{ID} = \sigma \oplus f_s(a)$

Figure 8.2: Weis, Sarma, Rivest, and Engels's protocol using a pseudo-random function

8.2 Feldhofer, Dominikus, and Wolkerstorfer

Feldhofer, Dominikus and Wolkerstorfer's protocol [73] differs from [178] in the fact that the pseudo-random function is replaced by AES. However, the general outline remains the same. The protocol is depicted in Figure 8.3.

Setup

Each tag is initialized with a randomly chosen secret key s, which is also stored by the system with the corresponding tag's identifier.

Interaction

The system picks a random number a and sends a request including a to the tag. Upon reception of this message, the tag picks a random number b and computes $\sigma = \mathrm{AES}_s(a, b)$, which is sent to the system.

Feldhofer *et al.* also proposed a variant of their 2-move protocol, where the authentication is mutual instead of being unilateral (Figure 8.4). In that case, when the system has identified the tag, that is to say when it has found the valid s, it computes and sends $\tau = \mathrm{AES}_s(b, a)$ to the tag. The tag checks that τ is a valid encryption of b and a, meaning that the reader is a legitimate one.

Search

Upon reception of σ, the system performs an exhaustive search in its database: for each entry ID, it computes $\mathrm{AES}_s^{-1}(\tau)$ until it finds a valid decryption.

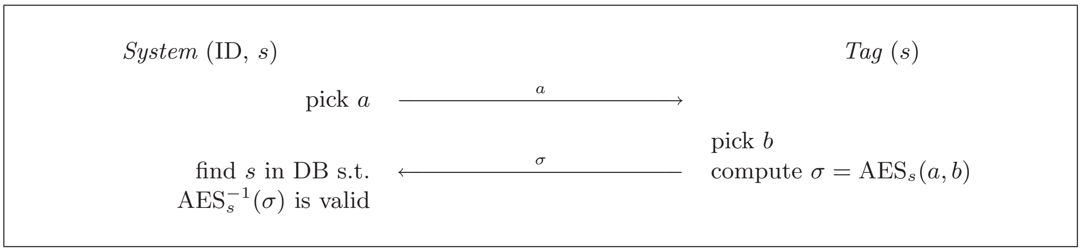

System (ID, s)
Tag (s)
pick a
a
pick b
find s in DB s.t.
σ
compute $\sigma = \mathrm{AES}_s(a, b)$
$\mathrm{AES}_s^{-1}(\sigma)$ is valid

Figure 8.3: Feldhofer, Dominikus, and Wolkerstorfer's authentication protocol

Note that [73] does not precise whether or not the same secret s is used for all tags. If it is the case, the tag must stores its identifier ID, which must be encrypted together with a and b. The paper is a bit ambiguous regarding this point.

8.3 Rhee, Kwak, Kim, and Won

Rhee, Kwak, Kim, and Won [151] proposed an authentication protocol that is based on a hash function h instead of AES. Their scheme, depicted in Figure 8.5, is defined as follows:

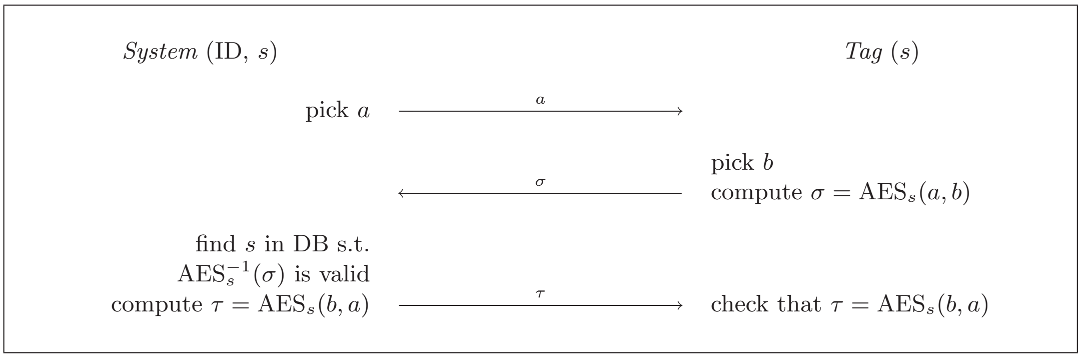

System (ID, s)
Tag (s)
pick a
a
pick b
compute $\sigma = \mathrm{AES}_s(a, b)$
σ
find s in DB s.t.
$\mathrm{AES}_s^{-1}(\sigma)$ is valid
compute $\tau = \mathrm{AES}_s(b, a)$
τ
check that $\tau = \mathrm{AES}_s(b, a)$

Figure 8.4: Feldhofer, Dominikus, and Wolkerstorfer's mutual authentication protocol

Setup

Each tag is initialized with a randomly chosen identifier ID. For each tag it manages, the system stores an entry in its database that contains this identifier.

Interaction

The system picks a nonce a and sends it to the tag. Upon reception of this message, the tag also picks a nonce b, computes $\sigma = h(\mathrm{ID}, a, b)$ and sends b and σ to the system. When the system has identified the tag, it computes and sends $\tau = h(\mathrm{ID}, b)$ to the tag. Given ID and b, the tag checks that τ is valid, meaning that the reader is a legitimate one. Thus, [151] ensures *mutual* authentication.

Search

Upon reception of σ, the system performs an exhaustive search in its database: for each entry ID, it computes $h(\mathrm{ID}, b)$ until it finds σ.

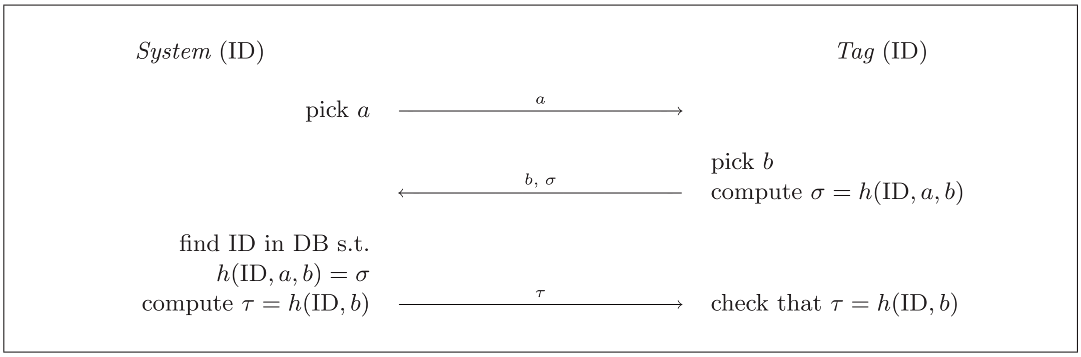

System (ID)
Tag (ID)
pick a
a
pick b
compute $\sigma = h(\mathrm{ID}, a, b)$
b, σ
find ID in DB s.t.
$h(\mathrm{ID}, a, b) = \sigma$
compute $\tau = h(\mathrm{ID}, b)$
τ
check that $\tau = h(\mathrm{ID}, b)$

Figure 8.5: Rhee, Kwak, Kim, and Won's mutual authentication protocol

8.4 Molnar and Wagner

The protocol suggested by Molnar and Wagner [136], depicted in Figure 8.6, relies on Weis *et al.*'s protocol [178]. However, the authors of [136] have noticed that [178] allows identification but not authentication: an adversary is able to query the tag and replay the session afterwards. Consequently, Molnar and Wagner put a nonce in the query of the reader. Furthermore, they add one move to their protocol in order to ensure mutual authentication. Thus, their protocol is similar to [151], except that a pseudo-random function is used instead of a hash function. The three phases of the protocol are described below.

Setup

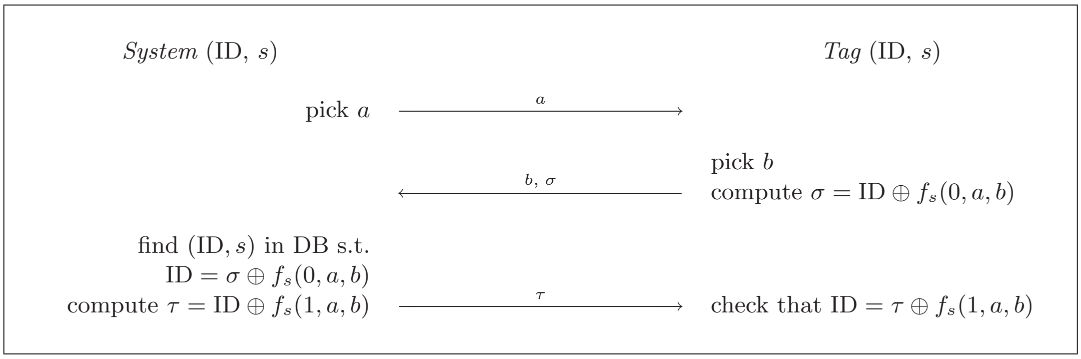
Each tag is initialized with a randomly chosen identifier ID and a randomly chosen secret s. For each tag it manages, the system stores an entry in its database that contains both ID and s.

Interaction

The system picks a nonce a and sends it to the tag. Upon reception of this message, the tag also picks a nonce b, computes $\sigma = \mathrm{ID} \oplus f_s(0, a, b)$ and sends b and σ to the system. When the system has identified the tag, it computes and sends $\tau = \mathrm{ID} \oplus f_s(1, a, b)$ to the tag. Given a, b, s, and ID, the tag checks that τ is valid, meaning that the reader is a legitimate one.

Search

Upon reception of σ, the system performs an exhaustive search in its database: for each entry (ID, s) it checks whether or not $\mathrm{ID} = \tau \oplus f_s(1, a, b)$.



System (ID, s)

Tag (ID, s)

pick a $\xrightarrow{a}$

$\xleftarrow{b,\ \sigma}$ pick b
compute $\sigma = \mathrm{ID} \oplus f_s(0, a, b)$

find (ID, s) in DB s.t.
$\mathrm{ID} = \sigma \oplus f_s(0, a, b)$
compute $\tau = \mathrm{ID} \oplus f_s(1, a, b)$ $\xrightarrow{\tau}$ check that $\mathrm{ID} = \tau \oplus f_s(1, a, b)$

Figure 8.6: Molnar and Wagner's mutual authentication protocol

8.5 Ohkubo, Suzuki, and Kinoshita

The protocol put forward by Ohkubo, Suzuki and Kinoshita [138] is fairly similar to the hash-version of [178]. The difference is that the hash is deterministic rather than randomized. In order to avoid traceability by an adversary, the identifier contained in the tag is consequently refreshed at each new identification, by using a second hash function. This bring an additional interesting property, which is the forward privacy as defined in Chapter 6. Thus, an adversary tampering with the tag is not able to trace it in the past. Clearly, this carries an extra cost in terms of computation as the tag has to calculate two hashes at each identification.

Setup

The tag personalization consists of storing in its memory a random identifier s^1, which is also recorded in the system database with the corresponding identifier ID. Two hash functions G and H are chosen.

Interaction

When the system queries the tag, it receives back $r^k = G(s^k)$ where s^k is the current identifier of the tag. While it is powered, the tag replaces s^k by $s^{k+1} = H(s^k)$.

Search

From r^k, the system has to identify the corresponding tag. In order to do this, it constructs the hash chains from each of the n initial values until it finds the expected r^k or until it reaches a given maximum limit m on the chain length. The lifetime of the tag is *a priori* limited to m identifications. However, when a tag is scanned by a legitimate reader, its field in the database can be refreshed. The threshold m is therefore the number of read operations on a single tag between two legitimate identification.

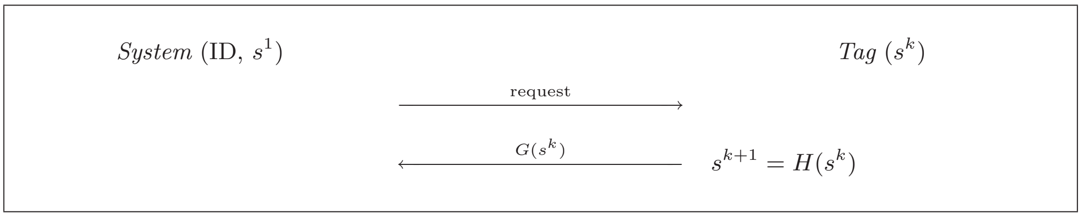


Figure 8.7: Ohkubo, Suzuki, and Kinoshita's identification protocol

The main advantage of this protocol compared to the previous challenge-response protocols is that it also assures forward secrecy. However, it does not prevent replay attacks. Common techniques to avoid replay attacks are usually incremental sequence number, clock synchronization, or a fresh challenge sent by the verifier. This latter option seems to be the most suited to our case. We proposed in [19] modifying Ohkubo, Suzuki, and Kinoshita's identification protocol into an authentication protocol, as depicted in Figure 8.8. Furthermore, we proposed to assure the reader authentication by adding a third message containing $G(s^{k+1} \oplus w)$ where w is a fixed and public non-zero binary string. A quite similar idea has been published recently by Dimitriou [64].

Note that Ohkubo, Suzuki, and Kinoshita modified their scheme in [139]. To reduce the complexity on both the system's side and tag's side, they propose not to apply the function H each time the tag is queried. Instead, they use a counter c and apply H only when the counter reaches its upper bound. Unfortunately, this technique degrades forward privacy because an adversary can trace the c last events of the tag (in the worst case) if she is able to tamper with it. Still worst, the value of the counter is sent to the reader each time the tag is queried and therefore it may be traced according to this value, which is not random.

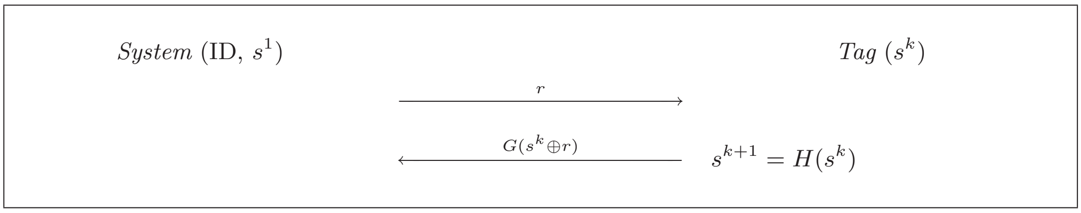


Figure 8.8: Ohkubo, Suzuki, and Kinoshita's modified protocol

8.6 Juels and Weis

8.6.1 Description of HB$^+$

In 2005, Weis [177] proposed a protocol that relies neither on hash functions nor on pseudo-random functions. Instead, his protocol only uses AND and XOR operations. It is based on the Hopper and Blum's secure human authentication protocol (HB) [104] that we describe below.

Consider a prover (which is the resource-constrained device) who shares with a verifier a ℓ-bit vector $\mathbf{x}$. At the beginning of the protocol, the verifier picks a random vector $\mathbf{a} \in \{0,1\}^\ell$ and sends it to the prover. The prover computes the binary inner product and sends the result back to the verifier, which can check whether or not the result is valid. A correct prover gives the right answer with probability 1 while a malicious prover cannot answer with a probability greater than one half. By iterating this process k times, the probability of success of a malicious prover cannot be better than 2^{-k}. With such a simple protocol, an adversary being able to eavesdrop $O(\ell)$ legitimate interactions between a prover and a verifier can recover the secret $\mathbf{x}$ using the Gaussian elimination. To thwart this attack, the prover injects noise into his response, i.e., he sends a wrong result with probability η where $\eta \in [0, 1/2[$. Thus, a passive adversary can no longer use the Gaussian elimination in order to recover the secret $\mathbf{x}$. This problem, known as the Learning Parity with Noise (LPN) problem, has been intensively studied for example in [36, 42, 43, 99, 104, 126, 132].

Unfortunately, HB is not resistant in presence of an active adversary. Indeed, obtaining ℓ equations with linearly independent $\mathbf{a}$ leads to the secret $\mathbf{x}$ using the Gaussian elimination. Juels and Weis [118] suggested a variant of HB called HB$^+$, which is resistant even in presence of an active adversary. The basic idea is, the prover also picks a random vector $\mathbf{b}$ that is sent to the verifier at the beginning of the protocol. The three phases of HB$^+$ are described below and depicted in Figure 8.9.

Setup

The tag personalization (prover) consists of storing in its memory and in the system database (verifier) two random binary vectors $\mathbf{a}$ and $\mathbf{b}$. The tag also contains a value $\eta \in [0, 1/2[$.

Interaction

When the tag is initially queried, the k-round authentication starts (note that this initial query does not appear in [118] and is therefore not depicted in Figure 8.9). At each round, the tag picks and sends a random vector $\mathbf{b}$ to the system, and receives back a random vector $\mathbf{a}$. From $\mathbf{a}$ and $\mathbf{b}$, it computes and sends the value $\sigma = (\mathbf{a} \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y}) \oplus \nu$ to the system, where ν is picked in $\{0, 1\}$ such that $\Pr(\nu = 1) = \eta$.

Search

At the end of the k rounds, the system considers that the authentication succeeds if less than ηk rounds failed, given that a round fails if and only if $\sigma \neq (\mathbf{a} \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y})$. Thus, for each entry in its database, the system must verify if the authentication succeeds.

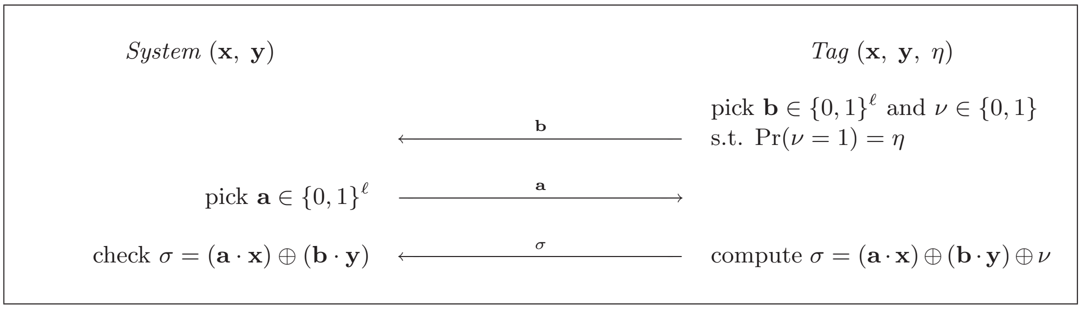

System $(\mathbf{x},\ \mathbf{y})$
Tag $(\mathbf{x},\ \mathbf{y},\ \eta)$
pick $\mathbf{b} \in \{0,1\}^\ell$ and $\nu \in \{0,1\}$
s.t. $\Pr(\nu = 1) = \eta$
$\mathbf{b}$
pick $\mathbf{a} \in \{0,1\}^\ell$
$\mathbf{a}$
check $\sigma = (\mathbf{a} \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y})$
σ
compute $\sigma = (\mathbf{a} \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y}) \oplus \nu$

Figure 8.9: One round of HB$^+$

Recently, Katz and Shin [120] announced that they are able to prove the security of HB (resp. HB$^+$) under concurrent executions. This would result in a HB/HB$^+$ protocol with only 2 (resp. 3) rounds instead of $O(k)$ rounds.

8.6.2 Attack on HB$^+$

In [118], Juels and Weis prove that their protocol is secure against an active adversary. By "active", they mean an adversary who is allowed to query the targeted tag, and only then she is allowed to interact with a legitimate reader once. This model implies that an adversary cannot access the reader when she queries her targeted tag, thus excluding man-in-the-middle attacks. This model is motivated by the fact that the adversary's goal as defined in [118] is to insert a counterfeit tag into the system without being detected.

Unfortunately, Gilbert, Robshaw, and Sibert [91] have exhibited a linear-time attack against HB$^+$ under a stronger adversary model. Their attack works when the adversary is able to carry out a man-in-the-middle attack between a legitimate reader and a legitimate tag. They also assume, as it was already supposed in [118], that the adversary is able to determine whether her attempt to mitigate the targeted tag succeeds or not. The attack consists in perturbating the challenge $\mathbf{a}$ sent by the reader by XORing it with a chosen ℓ-bit vector δ. This perturbating vector is re-used for each of the k rounds. If the overall authentication process succeeds, then we are almost certain that $\delta \cdot \mathbf{x} = 0$. If it fails then we are almost certain that $\delta \cdot \mathbf{x} = 1$. Repeating the attack ℓ times using linearly independent vectors δs enables recovering the secret vector $\mathbf{x}$.

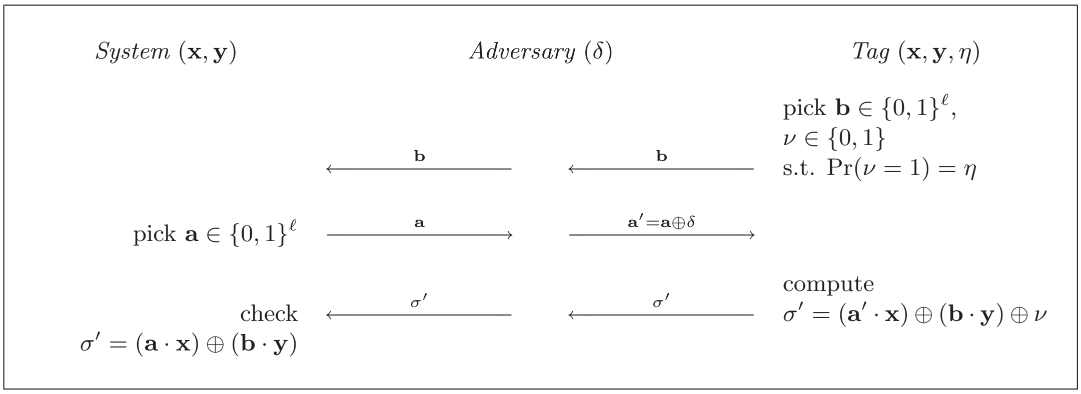

System $(\mathbf{x}, \mathbf{y})$
Adversary (δ)
Tag $(\mathbf{x}, \mathbf{y}, \eta)$
pick $\mathbf{b} \in \{0,1\}^\ell$,
$\nu \in \{0,1\}$
s.t. $\Pr(\nu = 1) = \eta$
$\mathbf{b}$
$\mathbf{b}$
pick $\mathbf{a} \in \{0,1\}^\ell$
$\mathbf{a}$
$\mathbf{a}' = \mathbf{a} \oplus \delta$
compute
$\sigma' = (\mathbf{a}' \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y}) \oplus \nu$
σ'
σ'
check
$\sigma' = (\mathbf{a} \cdot \mathbf{x}) \oplus (\mathbf{b} \cdot \mathbf{y})$

Figure 8.10: Gilbert, Sibert, and Robshaw's attack against HB$^+$

This attack is actually easily practicable. The adversary needs to be able to carry out a man-in-the-middle attack and to detect whether or not the authentication process succeeded. Such an attack could rely on an active relay attack, described in Chapter 6. Thus, it is undeniable that the adversary model used in [118] is not realistic in most applications.

8.7 Complexity Issues

In the previous sections, we have presented several protocols based on self-refreshment. They all suffer from a complexity issue: the system must carry out an exhaustive search over the identifiers it manages in order to identify one tag. This technique is quite expensive in terms of computation and cannot be used in large scale applications. This issue is mitigated in [118] because only AND and XOR operations are required, instead of hash functions or pseudo-random functions as in [73, 136, 138, 151, 178]. Unfortunately, as presented above, [118] is not secure in a realistic adversary model.

This complexity issue, which seems inherent to RFID protocols based on symmetric cryptography, is addressed in the next chapter.

Reducing Complexity in Radio Frequency Identification

CHAPTER NINE

Identification protocols based on symmetric challenge-response, denoted CR, assure privacy of the tag bearers, as shown in Chapter 8. Unfortunately, they always suffer from a large time complexity. Existing protocols require $O(n)$ cryptographic operations to identify one device among n and $O(n^2)$ in order to identify the whole system. Molnar and Wagner [136] have suggested a method to reduce the complexity to $O(\log n)$. We denote this technique by CR/MW and present it in Section 9.1. Later we show in Section 9.2 that it degrades privacy if the adversary has the possibility to tamper with at least one tag [19]. We evaluate precisely the threat according to the required time complexity. Because low cost devices are not tamper-resistant, such an attack could be feasible. In Section 9.3, we go thoroughly into the Ohkubo, Suzuki, and Kinoshita's protocol [138], denoted OSK, already introduced in Section 8.5. In Section 9.4, we propose an approach based on a time-memory trade-off whose goal is to improve OSK. This variant, called OSK/AO, is as efficient as CR/MW in practice but does not degrade privacy [19, 27]. Contrary to CR/MW, only the method used by the system to store its data is modified while CR/MW adapts the data stored by the tag and the data that are exchanged between the system and the tag. Thus the security of OSK/AO is equivalent to the security of OSK.

In order to illustrate our comparison between CR, CR/MW, OSK, and OSK/AO, we consider a real life scenario in a library, where tags are used to identify books. Several libraries already use this technology, for example the libraries of Santa Clara (USA), Heiloo (Netherlands), Richmond Hill (Canada), and K.U. Leuven (Belgium). Inside the library, tags make it possible to scan shelves for misfiled books and to identify books that have not been placed on the shelves. These tags also make the check-out and check-in of books much easier. When a user takes a book home, an adversary should not be able to find out what he is reading nor track him using the tags. Nor should the adversary be able to track him *a posteriori*, when the book has been brought back to the library. Indeed, the adversary could borrow the book and tamper with its tag to track the past events of the tag. In other words, the protocol should assure *forward privacy*. In a library scenario, it is realistic to assume that the tags can contain a secret-key cipher or a hash function because they are not disposable. Thus, a slightly higher cost is conceivable.

In the next sections, we assume that the system relies on a single computer which takes $\theta = 2^{-23}$ seconds to carry out a cryptographic operation. This value is rather arbitrary since it depends on the cryptographic building block itself, either an encryption function or a hash function. However, our goal is to choose a rather realistic value just to compare the protocols in a fair way, disregarding the underlying building blocks. We assume that inputs and outputs of the cryptographic functions are 128-bit long. The library manages 2^{20} tags. We assume that tag singulation and collision avoidance are private and performed at a lower layer as explained in Chapter 5. Identification of several tags is therefore sequential. Current implementations allow a single reader to read several hundreds of tags per second, meaning that the system should spend at the most a few milliseconds to identify one tag. In the following sections, t_{P} denote the average time to identify one tag using a protocol P. Because certain applications (in libraries, in amusement parks, etc.) may use numerous readers, the system should not become a bottleneck in terms of computation. Thus, the system should be capable of identifying the whole set of tags it manages in only a few seconds (e.g., for real-time inventories).

9.1 Molnar and Wagner's Protocol

9.1.1 Challenge-Response Building Block

The Molnar and Wagner's challenge-response building block (CR), depicted in Figure 9.1, provides mutual authentication of the reader and the tag in a private way. It prevents an adversary from impersonating, tracing or identifying tags.

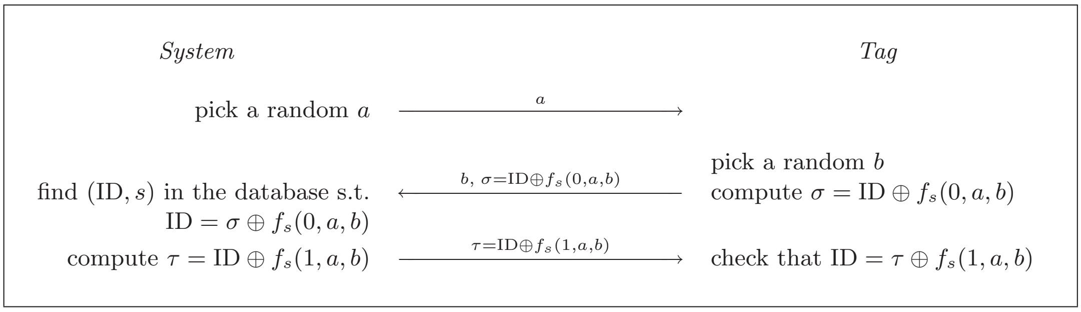

System
Tag
pick a random a
a
pick a random b
$b,\ \sigma = \mathrm{ID} \oplus f_s(0,a,b)$
find (ID, s) in the database s.t. $\mathrm{ID} = \sigma \oplus f_s(0,a,b)$
compute $\sigma = \mathrm{ID} \oplus f_s(0,a,b)$
compute $\tau = \mathrm{ID} \oplus f_s(1,a,b)$
$\tau = \mathrm{ID} \oplus f_s(1,a,b)$
check that $\mathrm{ID} = \tau \oplus f_s(1,a,b)$

Figure 9.1: Molnar and Wagner's challenge-response protocol

Let ID be the tag's identifier that is stored in both the system's database and the tag. They also share a secret key s. To initiate the authentication, the reader sends a nonce a to the tag. Subsequently, the tag picks a random b and answers $\sigma := \mathrm{ID} \oplus f_s(0,a,b)$, where f_s is a pseudo-random function. The system retrieves the tag's identity by finding the pair (ID, s) in its database such that $\mathrm{ID} = \sigma \oplus f_s(0,a,b)$. This completes the tag's authentication.

Now, in order to achieve mutual authentication, the system sends back $\tau := \mathrm{ID} \oplus f_s(1, a, b)$ to the tag. The tag can thus verify that the reader is authorized to read it by checking that $\mathrm{ID} = \tau \oplus f_s(1, a, b)$.

9.1.2 Efficiency

In order to identify a tag, the system must carry out an exhaustive search on the n secrets stored in its database. Therefore the system's workload is linear in the number of tags. More precisely, the average number of cryptographic operations required to identify one tag is $n/2$ and therefore we have $t_{\mathsf{CR}} = \frac{n\theta}{2}$. With $n = 2^{20}$ and $\theta = 2^{-23}$, we have $t_{\mathsf{CR}} \approx 62$ ms which is too high in practice. Since CR does not scale well in a system with many tags, we will now examine Molnar and Wagner's tree-based technique [136], whose main strength lies in the reduction of the system's workload from $O(n)$ to $O(\log n)$.

9.1.3 Tree-Based Technique

The technique suggested by Molnar and Wagner, called CR/MW, relies on a tree structure to reduce identification complexity. Instead of searching a flat space of secrets, let us arrange them in a balanced tree with branching factor δ. The tags are the leaves of this tree and each edge is associated with a value. Each tag has to store the values along the path from the root of the tree to itself. This sequence makes up its *secret*, and each value is called a *block of secret*. On the other side, the reader knows all the secrets. We describe the protocol below:

Setup

Let n be the number of tags managed by the system and $\ell := \lceil \log_\delta n \rceil$ be the depth of the tree with a branching factor δ. Each edge in the tree is valued with a randomly chosen secret $r_{i,j}$ where i is the level in the tree and j is the branch index. Figure 9.2 represents such a tree with parameters $n = 9$ and $\delta = 3$. The secret of a given tag is the list of the values $r_{i,j}$ from the root to the leaf. For example, the secret of T_5 in Figure 9.2 is $[r_{1,1}, r_{2,5}]$.

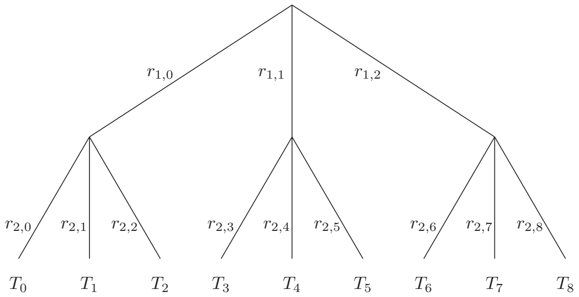


Figure 9.2: Tags' secrets tree

Interaction

The tag is queried level by level from the root to the leaves. At each level i, CR/MW runs CR for each secret of the explored subtree, that is, the reader tries every edge in turn to find out on which one the tag is. If CR fails for all current level's secrets, the tag rejects the reader and the protocol stops. If the reader has been successfully authenticated at each level, the protocol succeeds. Note that CR inevitably does not need to be performed δ times per level in practice. One run is enough if the reader checks the tag's answer with all current level's secrets, as described below.

Search

At each level i, the system has to search in a set of δ secrets for the one matching the tag's secret. Given that $[s_1, \ldots, s_\ell]$ denotes a secret, at level i, on an average, the system has to compute $f_{s_i}(0, a, b)$ $\delta/2$ times, implying that $\frac{\delta}{2}\ell$ operations are required to identify one tag. Thus we have

$$t_{\mathsf{CR/MW}} = \frac{\delta\theta}{2}\log_\delta n.$$

The identification of one tag is far below the threshold of a few milliseconds. Identifying the whole system would take more than 2 minutes when $\delta = 2^{10}$ and decreases to 2 seconds when $\delta = 2$. However, we will see in Section 9.2 that having a small branching factor enables tracing the tags.

9.1.4 Ramzan and Gentry Parallelization Technique

Ramzan and Gentry have suggested that certain underlying protocols allow all levels of the tree to be performed in parallel. This is the case with CR because the instances of the protocol are independent during the execution of CR/MW. This offers a nice solution to reduce the number of communication rounds in the tree protocol. Even if the communication complexity is not reduced, sending a long message could be better than a burst of short ones for short range radio communication. Note however that the exchange stops as soon as one step of the authentication fails, thus avoiding useless communication complexity. This is not the case with the Ramzan and Gentry technique where the communication complexity is constant.

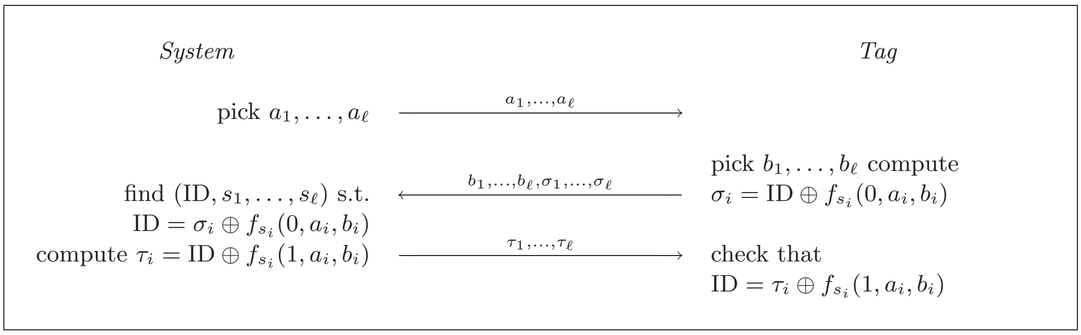

System
Tag
pick $a_1, \ldots, a_\ell$
$a_1, \ldots, a_\ell$
pick $b_1, \ldots, b_\ell$ compute
$\sigma_i = \mathrm{ID} \oplus f_{s_i}(0, a_i, b_i)$
$b_1, \ldots, b_\ell, \sigma_1, \ldots, \sigma_\ell$
find $(\mathrm{ID}, s_1, \ldots, s_\ell)$ s.t.
$\mathrm{ID} = \sigma_i \oplus f_{s_i}(0, a_i, b_i)$
compute $\tau_i = \mathrm{ID} \oplus f_{s_i}(1, a_i, b_i)$
$\tau_1, \ldots, \tau_\ell$
check that
$\mathrm{ID} = \tau_i \oplus f_{s_i}(1, a_i, b_i)$

Figure 9.3: Ramzan and Gentry's parallelization technique

9.2 Privacy-Weakening Attacks

9.2.1 Tampering With Only One Tag

In this section, we examine how the tree technique suggested by Molnar and Wagner allows tracing a tag when the adversary is able to tamper with some tag. The attack consists of three phases:

1. The adversary has one tag T_0 (e.g., her own) she can tamper with and thus obtain its complete secret. For the sake of calculation simplicity, we assume that T_0 is put back into circulation. When the number of tags in the system is large, this does not significantly affect the results.

2. She then chooses a target tag T. She can query it as much as she wants but she cannot tamper with it.

3. Given two tags T_1 and T_2 such that $T \in \{T_1, T_2\}$, the adversary can query T_1 and T_2 as many times as she wants but cannot tamper with them. We say that the adversary succeeds if she definitely knows which of T_1 and T_2 is T. We define the probability to trace T as being the probability that the adversary succeeds.

We assume that the underlying challenge-response protocol assures privacy when all the blocks of secrets are chosen according to a uniform distribution. We consequently assume that the adversary cannot carry out an exhaustive search over the secret space. Hence, the only way for an adversary to guess a block of secret of a given tag is to query it with the blocks of secret she obtained by tampering with some other tags. When she tampers with only one tag, she obtains only one block of secret per level in the tree. Thus, she queries T, and then T_1, and T_2 with this block. If either T_1 or T_2 (but not both) has the same block as T_0, she is able to determine which of them is T. If neither T_1 nor T_2 has the same block as T_0, she cannot answer. Finally, if both T_1 and T_2 have the same block as T_0, she cannot answer, but she can move onto the next level of the tree because the reader's authentication succeeded. We formalize the analysis below. We denote the secrets of T, T_0, T_1, and T_2 by $[s_1, \cdots, s_\ell]$, $[s_1^0, \cdots, s_\ell^0]$, $[s_1^1, \cdots, s_\ell^1]$, and $[s_1^2, \cdots, s_\ell^2]$ respectively. We consider a given level i where s_i^1 and s_i^2 are in the same subtree. Four cases must be considered:

- $C_i^1 = ((s_i^0 = s_i^1) \wedge (s_i^0 \neq s_i^2))$ then the attack succeeds,
- $C_i^2 = ((s_i^0 \neq s_i^1) \wedge (s_i^0 = s_i^2))$ then the attack succeeds,
- $C_i^3 = ((s_i^0 \neq s_i^1) \wedge (s_i^0 \neq s_i^2))$ then the attacks definitively fails,
- $C_i^4 = (s_i^0 = s_i^1 = s_i^2)$ then the attacks fails at level i but can move onto level $i+1$.

When the number of tags in the system is large, we can assume that

$$\Pr\left(C_i^1\right) = \Pr\left(s_i^0 = s_i^1\right) \times \Pr\left(s_i^0 \neq s_i^2\right).$$

The same assumption also applies to C_i^2, C_i^3, and C_i^4. Thus we have

$$\Pr\left(C_i^1 \vee C_i^2\right) = \frac{2(\delta - 1)}{\delta^2} \quad (1 \leq i \leq \ell)$$

and

$$\Pr\left(C_i^4\right) = \frac{1}{\delta^2}.$$

The overall probability P that the attack succeeds is therefore

$$\begin{aligned}
P &= \Pr\left(C_1^1 \vee C_1^2\right) + \sum_{i=2}^{\ell} \left(\Pr\left(C_i^1 \vee C_i^2\right) \times \prod_{j=1}^{i-1} \Pr\left(C_j^4\right) \right) \\
&= \frac{2(\delta-1)}{\delta^2} + \sum_{i=2}^{\ell} \left(\frac{2(\delta-1)}{\delta^2} \left(\frac{1}{\delta^2}\right)^{i-1} \right) \\
&= \sum_{i=1}^{\ell} \left(\frac{2(\delta-1)}{\delta^{2i}} \right) \\
&= 2(\delta-1) \frac{1-\left(\frac{1}{\delta^2}\right)^{\ell}}{1-\frac{1}{\delta^2}} \frac{1}{\delta^2}.
\end{aligned}$$

Given that $\delta^\ell = n$ we obtain

$$P = \frac{2}{\delta+1}\left(1 - \frac{1}{n^2}\right).$$

The curve of P when $n = 2^{20}$ is plotted in Figure 9.4.

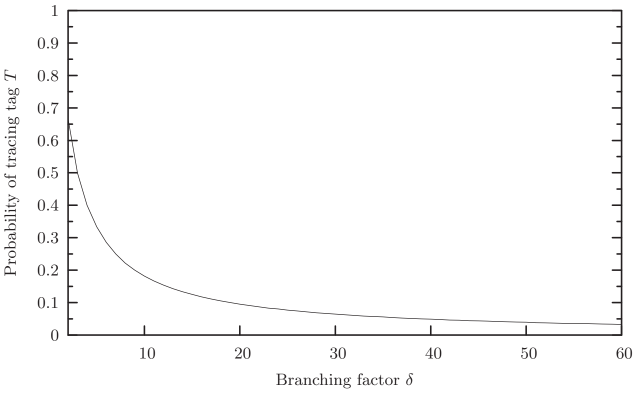


Figure 9.4: Probability of tracing a tag when the adversary tampered with only one tag

9.2.2 Tampering With Several Tags

We now consider the case where the adversary can tamper with several tags, e.g., she borrows several books in the library in order to tamper with their tags. We examine the influence of the number of opened tags on the probability of tracing the target tag. As before, each opened tag is put back into circulation to simplify calculations. When n is large, this does not significantly affect the results. As in the previous section, we denote the secrets of T, T_1, and T_2 by $[s_1, \cdots, s_\ell]$, $[s_1^1, \cdots, s_\ell^1]$, and $[s_1^2, \cdots, s_\ell^2]$ respectively. We consider a given level i where s_i^1 and s_i^2 are in the same (one-level) subtree. Let $\mathcal{K}_i$ denote the set of blocks of this (one-level) subtree that are known by the adversary and let $\mathcal{U}_i$ denote the set of those that are unknown by the adversary. k_i denotes the number of blocks in $\mathcal{K}_i$. Five cases must be considered:

- $C_i^1 = ((s_i^1 \in \mathcal{K}_i) \wedge (s_i^2 \in \mathcal{U}_i))$ then the attack succeeds,
- $C_i^2 = ((s_i^1 \in \mathcal{U}_i) \wedge (s_i^2 \in \mathcal{K}_i))$ then the attack succeeds,
- $C_i^3 = ((s_i^1 \in \mathcal{K}_i) \wedge (s_i^2 \in \mathcal{K}_i) \wedge (s_i^1 \neq s_i^2))$ then the attack succeeds,
- $C_i^4 = ((s_i^1 \in \mathcal{U}_i) \wedge (s_i^2 \in \mathcal{U}_i))$ then the attacks definitively fails,
- $C_i^5 = ((s_i^1 \in \mathcal{K}_i) \wedge (s_i^2 \in \mathcal{K}_i) \wedge (s_i^1 = s_i^2))$ then the attacks at level i fails but can move onto level $i+1$.

Thus, we have for all i such that $1 \leq i \leq \ell$:

$$\begin{aligned}\Pr(C_i^1 \vee C_i^2 \vee C_i^3) &= \frac{2k_i}{\delta}\left(1-\frac{k_i}{\delta}\right)+\left(\frac{k_i}{\delta}\right)^2\left(1-\frac{1}{k_i}\right)\\ &= \frac{k_i}{\delta^2}(2\delta-k_i-1),\end{aligned}$$

and

$$\Pr(C_i^5) = \frac{k_i}{\delta^2}.$$

The overall probability P that the attack succeeds is therefore

$$\begin{aligned}P &= \Pr(C_1^1 \vee C_1^2 \vee C_1^3) + \sum_{i=2}^{\ell}\left(\Pr\left(C_i^1 \vee C_i^2 \vee C_i^3\right) \times \prod_{j=1}^{i-1}\Pr\left(C_j^5\right)\right)\\ &= \frac{k_1}{\delta^2}\left(2\delta-k_1-1\right) + \sum_{i=2}^{\ell}\left(\frac{k_i}{\delta^2}(2\delta-k_i-1)\prod_{j=1}^{i-1}\frac{k_j}{\delta^2}\right).\end{aligned}$$

We now compute k_1, i.e., the number of different blocks known by the adversary at level 1, given that k_0 is the number of tags tampered with by the adversary. We have

$$k_1 = \delta\left(1-(1-\frac{1}{\delta})^{k_0}\right)$$

and then

$$k_i = \delta\left(1-(1-\frac{1}{\delta})^{g(k_i)}\right) \quad (2 \leq i \leq \ell),$$

where

$$g(k_i) = k_0\prod_{j=1}^{i-1}\frac{1}{k_j}.$$

The probability that the attack succeeds, according to the branching factor δ and given that k_0 tags have been opened, is plotted in Figure 9.5 with 2^{20} tags in the system. We would like to highlight the surprising behavior of P when the branching factor is small. This is due to the fact that neither $\Pr(C_i^1 \vee C_i^2 \vee C_i^3)$ nor $\Pr(C_i^5)$ are monotonous and they reach their optimum at different values of δ. Table 9.1 supplies a few values in order to illustrate our attack.

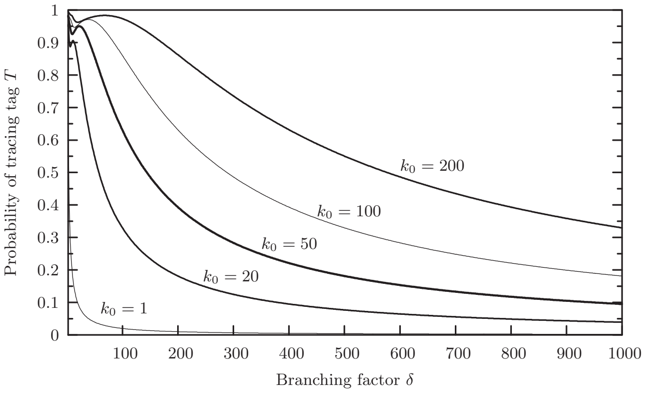


Figure 9.5: Probability of tracing a tag when the adversary tampered with k_0 tags

δ k_0	2	20	100	500	1000
1	66.6%	9.5%	1.9%	0.3%	0.1%
20	95.5%	83.9%	32.9%	7.6%	3.9%
50	98.2%	94.9%	63.0%	18.1%	9.5%
100	99.1%	95.4%	85.0%	32.9%	18.1%
200	99.5%	96.2%	97.3%	55.0%	32.9%

Table 9.1: Probability that the attack succeeds

9.2.3 Notes on the Original Tree-Based Technique

In the original tree-based scheme proposed by Molnar and Wagner in [136], the blocks of secret of the tags were not chosen according to a uniform distribution. Instead, subtrees of a given level had the same set of blocks of secrets. This seems to be due to a typo in the setup algorithm of [136].

The attack is obviously more efficient on this original scheme because, the k_is are larger (for a same value of k_0) than in the scheme analyzed above. Moreover, parallelizing all levels of the tree using the Ramzan and Gentry technique further threatens the privacy in that case because the blocking property of the authentication is lost. In other words, if the adversary fails at level i, she can still try to trace the tags at level $i+1$ with the set of blocks of secret she has. The problem arises from the fact that that tag cannot verify the reader's identity before

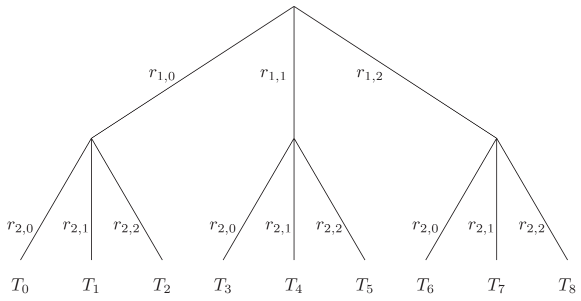


Figure 9.6: Tree of tags' secrets in the original scheme

having sent its full message. To retrieve the blocking behavior of the tags, the parallelization should result in a protocol that provides no useful information at level i to a reader that failed at any level $j < i$. We propose a protocol that meets these requirements. The basic principle is to link together all secret blocks by XORing them. Instead of sending $\sigma_i = \mathrm{ID} \oplus f_{s_i}(0, \cdot, \cdot)$ at level i, the tag sends $\sigma_i = \mathrm{ID} \oplus f_{s_1 \oplus \cdots \oplus s_i}(0, \cdot, \cdot)$. The messages exchanged between the reader and the tag are depicted in Figure 9.7.

System		*Tag*
pick $a_1, \ldots, a_\ell$	$\xrightarrow{a_1,\ldots,a_\ell}$	
find $(\mathrm{ID}, s_1, \ldots, s_\ell)$ s.t. $\mathrm{ID} = \sigma_i \oplus f_{s_1 \oplus \cdots \oplus s_i}(0, a_i, b_i)$	$\xleftarrow{b_1,\ldots,b_\ell,\sigma_1,\ldots,\sigma_\ell}$	pick $b_1, \ldots, b_\ell$ compute $\sigma_i = \mathrm{ID} \oplus f_{s_1 \oplus \cdots \oplus s_i}(0, a_i, b_i)$
compute $\tau_i = \mathrm{ID} \oplus f_{s_1 \oplus \cdots \oplus s_i}(1, a_i, b_i)$	$\xrightarrow{\tau_1,\ldots,\tau_\ell}$	check that $\mathrm{ID} = \tau_i \oplus f_{s_1 \oplus \cdots \oplus s_i}(1, a_i, b_i)$

Figure 9.7: Ramzan and Gentry's improved parallelization technique

9.3 Ohkubo, Suzuki, and Kinoshita's Protocol

9.3.1 Description

In this section, we briefly recall Ohkubo, Suzuki, and Kinoshita's protocol [138], already introduced in Section 8.5.

Setup

The personalization of a tag T_i consists of storing in its memory a random identifier s_i^1, which is also recorded in the database of the system. Thus, the database initially contains the set of random values $\{s_i^1 \mid 1 \leq i \leq n\}$. Two hash functions G and H are chosen. One hash function is enough if a one-bit parameter is added to the function.

Interaction

When the system queries T_i, it sends an identification request to the tag and receives back $r_i^k := G(s_i^k)$ where s_i^k is the current identifier of T_i. While T_i is powered, it replaces s_i^k by $s_i^{k+1} := H(s_i^k)$. The exchanges between the system and the tag can be represented as follows:

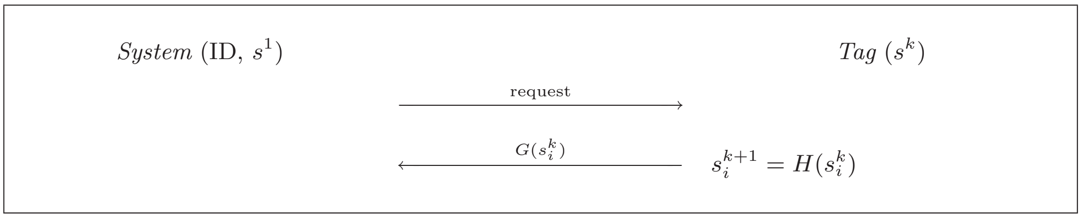


Figure 9.8: Ohkubo, Suzuki, and Kinoshita's protocol

Search

From r_i^k, the system has to identify the corresponding tag. In order to do this, it constructs the hash chains from each n initial value s_i^1 until it finds the expected r_i^k or until it reaches a given maximum limit m on the chain length.

9.3.2 Replay Attack Avoidance and Reader Authentication

Like CR, OSK assures privacy because the information sent by the tag is indistinguishable from a random value, in the random oracle model. The main advantage of OSK compared to CR is that it assures forward privacy, meaning that if an adversary can tamper with a tag, she is not able to track its past events. However, OSK does not prevent replay attacks. Common techniques to avoid replay attacks are incremental sequence number, clock synchronization, or a nonce. This latter option is the most suited to RFID tags. Thus we propose modifying OSK as depicted in Figure 9.9. Note that OSK does not provide reader authentication. However, this feature can be obtained if the system sends a third message containing $G(s_i^{k+1} \oplus w)$ where w is a fixed and public non-zero binary string, as depicted in Figure 9.10

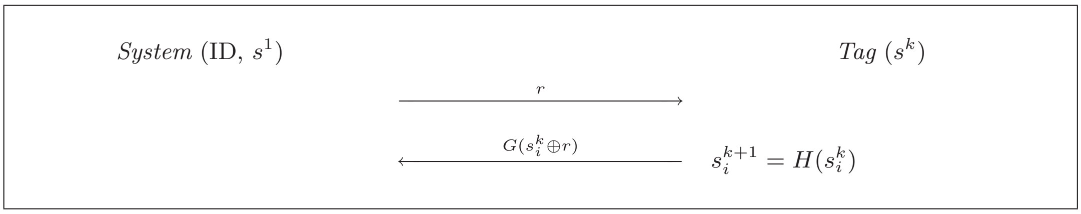


Figure 9.9: Modified OSK thwarting replay attacks

9.3.3 Efficiency

Outside the library, tags can be queried by foreign readers. This avoids maintaining synchronization between the tag and the system. Therefore, the complexity in terms of hash

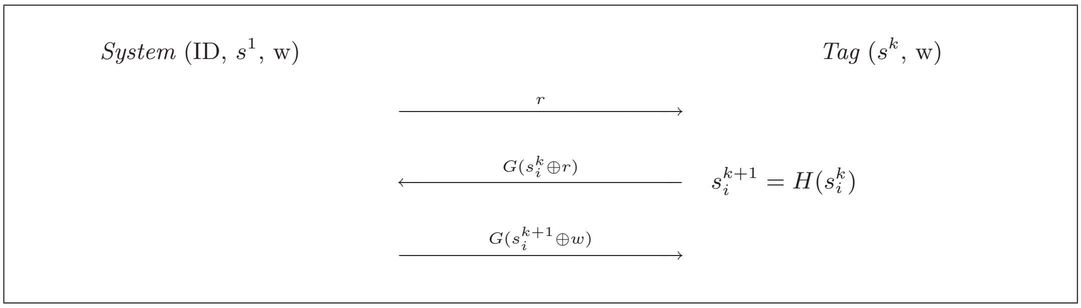


Figure 9.10: Modified OSK with mutual authentication

operations in order to identify one tag is $t_{\text{OSK}} = mn\theta$ on average (2 hash operations are carried out $mn/2$ times). With the parameters $n = 2^{20}$, $\theta = 2^{-23}$, and chains of length $m = 128$, we have $t_{\text{OSK}} \approx 16$ seconds. Note that if we had considered that readers of the library may read foreign tags (held by people in the library), then the complexity would tend towards to $2mn$ because the system would have to explore the whole database to determine whether or not a tag is owned by the system. Note that even if tags and readers were able to stay synchronized, for example when the RFID system is deployed in a closed environment, the complexity of OSK cannot be better than CR if no additional memory is used.

9.4 Improving OSK Using Time-Memory Trade-Off

In this section we provide a technique [19, 27] whose goal is to reduce the complexity of OSK. In order to do that, we briefly recall the required background on time-memory trade-off. More details on time-memory trade-off are available in Chapter 10 which is fully devoted to this topic.

9.4.1 Time-Memory Trade-Off

To reduce the complexity of OSK, we propose improving how secrets are managed by the system, without modifying the exchanges between tags and reader. For that, we suggest using a specific time-memory trade-off based on Hellman's original work [100] and Oechslin's optimizations [137]. This type of trade-off reduces the amount of work T needed to invert any given value in a set of N outputs of a one-way function E with the help of M units of memory. The efficiency follows the rule $T = N^2\gamma/M^2$ where γ is a small factor depending on the probability of success and the particular type of trade-off being used (see [23]). Compared to a brute-force attack, the trade-off can typically reduce the amount of work from N to $N^{2/3}$ using $N^{2/3}$ units of memory.

The basic idea of time-memory trade-off techniques consists in chaining (almost) all the possible outputs of E using a *reduction function* R that generates an arbitrary input of E from one of its outputs. By alternating E and R on a chosen initial value, a chain of inputs and outputs of E can be built. If enough chains of a given length are generated, most outputs of E will appear at least once in any chain. The trade-off comes from the fact that only the first and the last element of each chain is stored. Thus, a substantial memory space is saved, but computations will be required on-the-fly to invert a given element. Given one output

r of E that should be inverted, a chain starting at r is generated. If r was part of any stored chain, the last element of a chain in the table will eventually be reached. Looking up the corresponding start of the chain, we can regenerate the complete chain and find the input of E that yields the given output r. To assure a high success rate, several tables have to be generated with different reduction functions. The exact way of doing this is what differentiates existing trade-off schemes.

In what follows, we use *perfect rainbow tables* [137] that have been shown to perform better than other types of tables. The characteristic of the rainbow tables is each column of a table having a different reduction function. So, when two chains collide, they do not merge (except if they collide at the same position in the chain). When the residual merged chains are removed during the precomputation step, the tables are said to be perfect. With such tables and a probability of success of 99.9%, we have $\gamma = 8$.

9.4.2 Adapting the Trade-Off to Our Case

The time-memory trade-off technique described above cannot be directly applied to our case. Indeed, the input of E must cover all the identifiers but no more. Otherwise, the system would have no advantage over the adversary. Consequently, it is important to choose E such that its input space is as small as possible. We define the function E as follows:

$$E : (i, k) \mapsto r_i^k = G(H^{k-1}(s_i^1))$$

where $1 \leq i \leq n$ and $1 \leq k \leq m$. Thus, given the tag number and the identification number, E outputs the value which will be sent by the tag. We need also to define an arbitrary reduction function R such that

$$R : r_i^k \mapsto (i', k')$$

where $1 \leq i' \leq n, 1 \leq k' \leq m$. For example, we take

$$R(r) = (1 + (r \bmod n), 1 + (\left\lfloor \frac{r}{n} \right\rfloor \bmod m)).$$

There are still two important points that distinguish common time-memory trade-off from ours.

Firstly, the brute force method of OSK needs $n|s|$ units of memory to store the n values s_i^1 while usual brute-force methods do not require any memory. Thus, it makes sense to measure the amount of memory needed by the trade-off in multiples of $n|s|$. We call c the ratio between the memory used by the trade-off and the memory used by the brute-force. The memory used to store the tables is a multiple of the size of a chain while it is a multiple of s in the case of the brute-force. A stored chain is represented by its start and end point that can either be the output of E or its input. In our case the input is smaller, we therefore choose to store two pairs of (i, k), thus requiring $2(|n| + |m|)$ bits of memory. The conversion factor from units of brute-force to units of trade-off is $\mu = |s|/(2|n| + 2|m|)$. In the scenarios we are interested in, μ is typically between 2 and 4.

Secondly, when used in the trade-off, E is more complex than when used in the brute-force. Indeed, in the brute-force, the hash chains are calculated sequentially, thus needing just one H and one G calculation at each step. In the trade-off, i and k are arbitrary results from R and have no incremental relation with previous calculations. Thus, on average, each step computes the function E $(m-1)/2+1$ times and the function G once. We can now rewrite the trade-off relation as:

$$\begin{aligned} T &= \frac{N^2}{M^2}\gamma \\ &= \frac{n^2m^2}{(c-1)^2\mu^2n^2}(\frac{m-1}{2}+1)\gamma \\ &\approx \frac{m^3\gamma}{2(c-1)^2\mu^2}. \end{aligned}$$

We now show how this issue can be mitigated. So far, among the c shares of memory, $(c-1)$ shares are used to store the chains, and 1 share is used to store the n values s_i^1. If we not only store the first element of the chains, but also the element at the middle of the chain, we sacrifice even more memory but we reduce the average complexity of E. We will have only $(c-2)$ shares of the memory available for the tables, but E will have a complexity of $\frac{m-1}{4}+1$ (we need to generate only a quarter of a chain on average). We therefore have a trade-off between the memory sacrificed to store the intermediary points and the complexity of E. In general, if we store x values per chain, sacrificing x shares of memory, the trade-off complexity becomes:

$$\begin{aligned} T &= \frac{n^2m^2}{(c-x)^2\mu^2n^2}\left(\frac{m}{2x}+1\right)\gamma \\ &\approx \frac{m^3\gamma}{2x(c-x)^2\mu^2}. \end{aligned}$$

The optimal complexity is achieved when $x=\frac{c}{3}$. So we have

$$T_{\text{optimal}} \approx \frac{3^3}{2^3}\frac{m^3\gamma}{c^3\mu^2}. \tag{9.1}$$

Since a pair of (i,k) is 27 bits large (20 bits for i and 7 bits for k) we need at most 54 bits to store one chain. Thus, in the same amount of memory we would require to store one s $(\mu \geq 2)$, we could actually store more than two chains. From Equation 9.1, we can compute the time required to identify one tag. Assuming that all calculations are carried out on a single back-end equipped with $\frac{c(n|s|)}{8}=2^{24}c$ bytes of memory, and that we choose a success rate of 99.9% $(\gamma=8)$, the time to read a tag is

$$t_{\mathsf{OSK/AO}} \approx \frac{6^9\theta}{c^3} \text{ seconds.}$$

Figure 9.11 shows the identification time according to the available memory. For example, with 1 GB of RAM (i.e., c=64), we have $t_{\mathsf{OSK/AO}} \approx 0.004$ milliseconds.

Note that the time-memory trade-off cannot be applied directly to the modified OSK suggested in Section 9.3.2. This is due to the randomization of the tag's answer. In order to applied our time-memory technique on the modified version of OSK, the tag must answer with both $G(s_i^k)$ and $G(s_i^k \oplus r)$. The former value enables the reader to identify the tag and the latter one allows to detect replay attacks.

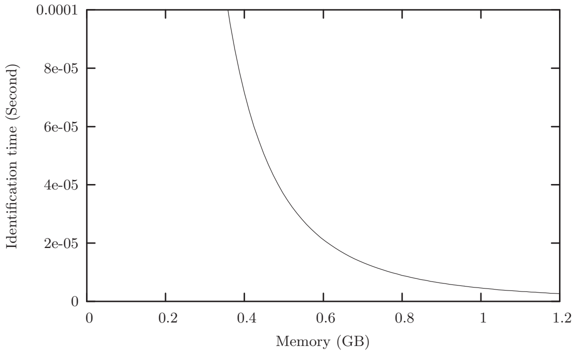


Figure 9.11: OSK/AO time complexity

9.4.3 (P)recomputation of the Tables

Before the trade-off can be used to read the tags, the trade-off chains must be generated and their start and end stored in the memory. Since the chains contain arbitrary hashes, we need to generate slightly more than nm hashes to ensure that each hash appears at least once in the tables with a high probability. Again, the hashes are not created sequentially and each calculation of E incurs about $\frac{m}{2}+1$ hash calculations. The effort to create the tables is thus $T_{precalc} \approx nm^2/2$. This complexity is reduced by the fact that we store intermediate elements of the chains is some part of the memory.

If the set of tags in the system stays the same, the tables only need to be calculated once. If new tags must be added, the tables must be recalculated. Extra tags can be included in the tables, so that they need not be recalculated for every single new tag. Every time the tables are recalculated we can also remove tags that are no longer in use. Typically the tables could be recalculated off-line every night, week or month.

Keeping m low increases the advantage of the trade-off over the brute-force method. The following procedure can be applied to keep m small. In the database that contains the s_i^1 we can keep track of how many times each tag was read. We know that the next time we read the tag, the result will be further down the hash chain. If tag i has been read k times, we can thus replace s_i^1 by s_i^k in the database when the next recalculation of the tables occurs. Thus m is no longer the number of times a tag is read in its lifetime but the maximum number of times it is read between two recalculations of the tables, or the maximum number of times it is read by a foreign reader. Note that the adjustment of s_i^1 makes both the trade-off and the brute-force method faster but increases the speed-up factor between the two.

Time-memory trade-offs are probabilistic, thus there is an arbitrarily small chance that a tag may not be found in the tables because a particular s_i^k is not part of any chain that was generated. A pragmatic approach to this problem is simply to read the tag a second time in such a case (hoping that s_i^{k+1} will be found). A more deterministic approach would be to keep score of the hash values that are generated when the tables are calculated and to eliminate the s_i^1 for which not all hash values have appeared.

Given that $n = 2^{20}$, $m = 2^7$, and $\theta = 2^{-23}$, in our library example, the precomputation takes about 17 minutes.

9.5 Comparison

Below, we summarize our analysis of the CR, CR/MW, OSK, and OSK/AO protocols.

Firstly, we consider the storage aspect. On the tag side, the storage of the identifiers becomes a real problem with CR/MW when the branching factor δ is small. Having a large δ is therefore preferable. Storage is the main drawback of OSK/AO because precomputation and storage of tables is required. In the example given in Section 9.4, 1 GB of RAM is used. Today, such a memory is available on Joe Blow's computer.

Next, we address the complexity question. Both CR/MW and OSK/AO are parameterizable. CR/MW depends on δ which can be chosen between 2 and n. Obviously, the case $\delta = n$ leads to CR. Having $\delta > \sqrt{n}$ is possible but in this case the tree is no longer complete. In fact, a typical value could be $\delta = \sqrt{n}$. On the other hand, OSK/AO depends on the available memory. Table 9.2 gives a numerical comparison of CR, CR/MW, OSK, and OSK/AO.

Scheme (parameter)	Time (millisecond)
CR	62.500
CR/MW ($\delta = 2^{10}$)	0.122
CR/MW ($\delta = 2$)	0.002
OSK	16'000.000
OSK/AO (342 MB)	0.122
OSK/AO (1.25 GB)	0.002

Table 9.2: Time to identify one tag

We now consider the privacy issue. While CR is secure, CR/MW degrades the privacy because, when an adversary is able to tamper with at least one tag (e.g., her own tag), she is also able to trace other tags in a probabilistic way. We have shown that the probability to trace tags decreases when the computation complexity grows. Thus, CR/MW can be seen as a trade-off between privacy and complexity. We proved that the probability to trace tags is far from being negligible. For example, when the branching factor is $\delta = 2^{10}$, the probability to trace a tag is about 0.1% when only one tag has been opened, but it is about 32.9% when 200 tags have been opened (refer Table 9.1). OSK/AO inherits from the security proofs of OSK, in particular the fact that OSK is forward private, because it modifies neither the information exchanged, nor the tag's content. It only improves the way the system manages and stores the data.

Thus, we can say that the main advantage of CR/MW lies in the fact that it does not require precomputation while the advantage of OSK/AO is that it remains secure.The number of tag readings with OSK/AO is limited by the chain length while it is not with CR/MW. However, crossing this threshold does not threaten privacy. Finally, when CR/MW is used, we recommend using a large branching factor in order to limit the privacy threat.

False Alarm Detection in Time-Memory Trade-Offs

CHAPTER TEN

Having used time-memory trade-offs in the previous chapter to reduce complexity in RFID systems, we analyzed ways to improve this technique, originally proposed by Hellman [100].

Since Hellman's original publication, a few optimizations of the method have been suggested. For all these trade-off variants, the cryptanalysis time is reduced by the square of the available memory. One fraction of the cryptanalysis work is due to the so-called "false alarms". In typical applications, false alarms can incur half of the work or even more. We propose a false alarm detection method [23, 24] that can significantly reduce the work due to these false alarms by using a small amount of memory. Our method can reduce the time by much more than the square of the additional memory used. In the example we provide, an increase of 2% of memory yields a 10% increase in performance.

In Chapter 9, we have already dealt with time-memory trade-offs. In Section 10.1, we give a more detailed approach and we recap the variants that have been proposed during the last decades. Then, in Section 10.2, we introduce a new concept so-called *checkpoint*. We propose a technique based on checkpoints that enables reducing the time spent to detect false alarms. Our method works with the classic trade-off, with distinguished points, and with rainbow tables. In Section 10.2, we give a rough idea of our technique. In Section 10.3, we provide a detailed and formal analysis of the rainbow tables. These new results allow to better understand the rainbow tables and to formally compute the probability of success, the computation time, and the optimal size of the tables. Based on this analysis we can describe and evaluate our technique in detail. We experiment it on Windows password cracking (based on DES) as proposed by Oechslin [137] in 2003. Finally, in Section 10.5, we describe further research, explaining how the checkpoints could be used during the precomputation phase in order to reduce the amount of memory needed to store the chains.

GILDAS AVOINE

10.1 History of Trade-Offs

Many cryptanalytic problems can be solved in theory using an exhaustive search in the key space, but are still hard to solve in practice because each new instance of the problem requires restarting the process from scratch. The basic idea of a time-memory trade-off is to carry out an exhaustive search once for all such that following instances of the problem become easier to solve. Thus, if there are N possible solutions to a given problem, a time-memory trade-off can solve it with T units of time and M units of memory. In the methods we are looking at, T is proportional to N^2/M^2 and a typical setting is $T = M = N^{2/3}$.

The cryptanalytic time-memory trade-off has been introduced in 1980 by Hellman [100] and applied to DES. Given a plaintext P and a ciphertext C, the problem consists in recovering the key K such that $C = \mathsf{E}_K(P)$, where E is an encryption function assumed to follow the behavior of a random function. Encrypting P under all possible keys and storing each corresponding ciphertext allows for immediate cryptanalysis but needs N elements of memory. The idea of a trade-off is to use chains of keys. It is possible thanks to a reduction function R that generates a key from a ciphertext. Using E and R, chains of alternating ciphertexts and keys can thus be generated. The key point is that only the first and the last element of each chain are stored. In order to retrieve K, a chain is generated from C. If at some point it yields a stored end of chain, then the entire chain is regenerated from its starting point. However, finding a matching end of chain does not necessarily imply that the key will be found in the regenerated chain. There exist situations where the chain that has been generated from C merges with a chain that is stored in the memory that does not contain K. This situation is called a *false alarm*. Matsumoto, with Kusuda [124] in 1996 and with Kim [121] in 1999, gave a more precise analysis of the trade-off parameters. In 1991, Fiat and Naor [74, 75] showed that there exist cryptographically sound one-way functions that cannot be inverted with such a trade-off.

Since the original work of Hellman, several improvements have been proposed. In 1982, Rivest [61] suggested an optimization based on *distinguished points* (DP) which greatly reduces the amount of look-up operations that are needed to detect a matching end point in the table. Distinguished points are keys (or ciphertexts) that satisfy a given criterion, e.g., the last n bits are all zero. In this variant, chains are not generated with a given length but they stop at the first occurrence of a distinguished point. This greatly simplifies the cryptanalysis. Indeed, instead of looking up in the table each time a key is generated on the chain from C, keys are generated until a distinguished point is found and only then a look-up is carried out in the table. If the average length of the chains is t, this optimization reduces the amount of look-ups by a factor t. Because merging chains significantly degrades the efficiency of the trade-off, Borst, Preneel, and Vandewalle [47] suggested in 1998 to clean the tables by discarding the merging and cycling chains. This new kind of tables, called *perfect table*, substantially decreases the required memory. Later, Standaert, Rouvroy, Quisquater, and Legat [163] dealt with a more realistic analysis of distinguished points and also proposed an FPGA implementation applied to DES with 40-bit keys. Distinguished points can also be used to detect collisions when a function is iterated, as proposed by Quisquater and Delescaille [145], and van Oorschot and Wiener [180].

In 2003, Oechslin [137] introduced the trade-off based on *rainbow tables* and demonstrated the efficiency of his technique by recovering Windows passwords. A rainbow table uses a different reduction function for each column of the table. Thus two different chains can merge only if they have the same key at the same position in the chain. This makes it possible to generate much larger tables. Actually, a rainbow table acts almost as if each column of the table was a separate single classic[1] table. Indeed, collisions within a classic table (or a column of a rainbow table) lead to merges whereas collisions between different classic tables (or different columns of a rainbow table) do not lead to a merge. This analogy can be used to demonstrate that a rainbow table of mt chains of length t has the same success rate as t single classic tables of m chains of length t. As the trade-off is based on distinguished point, rainbow tables reduce the amount of look-ups by a factor of t, compared to the classic trade-off. Up until now, trade-off techniques based on rainbow tables are the most efficient ones. Recently, an FPGA implementation of rainbow tables has been proposed by Mentens, Batina, Preneel, and Verbauwhede [131] to retrieve Unix passwords.

10.2 Introduction to Checkpoints

10.2.1 False Alarms

When the precomputation phase is achieved, a table containing m starting points $S_1, \dots, S_m$ and m end points $E_1, \dots, E_m$ is stored in memory. This table can be regenerated by iterating the function f, defined by $f(K) := \mathsf{R}(\mathsf{E}_K(P))$, on the starting points. Given a row j, let

$$X_{j,i+1} := f(X_{j,i})$$

be the i-th iteration of f on S_j and let

$$E_j := X_{j,t}.$$

We have:

$$\begin{array}{ccccccccccc}
S_1 & = & X_{1,1} & \xrightarrow{f} & X_{1,2} & \xrightarrow{f} & \dots & \xrightarrow{f} & X_{1,t} & = & E_1 \\
S_2 & = & X_{2,1} & \xrightarrow{f} & X_{2,2} & \xrightarrow{f} & \dots & \xrightarrow{f} & X_{2,t} & = & E_2 \\
 & \vdots & & & & & & & & \vdots & \\
S_m & = & X_{m,1} & \xrightarrow{f} & X_{m,2} & \xrightarrow{f} & \dots & \xrightarrow{f} & X_{m,t} & = & E_m
\end{array}$$

In order to increase the probability of success, i.e., the probability that K appears in the stored values, several tables with different reduction functions are generated.

Given a ciphertext $C = \mathsf{E}_K(P)$, the on-line phase of the cryptanalysis works as follows: R is applied on C in order to obtain a key Y_1, and then the function f is iterated on Y_1 until matching any E_j. Let s be the length of the generated chain from Y_1:

$$C \quad \xrightarrow{\mathsf{R}} \quad Y_1 \quad \xrightarrow{f} \quad Y_2 \quad \xrightarrow{f} \quad \dots \quad \xrightarrow{f} \quad Y_s$$

Then the chain ending with E_j is regenerated from S_j until it yields the expected key K. Unfortunately in most of the cases K is not in the explored chain. Such a case occurs when R collides: the chain generated from Y_1 merged with the chain regenerated from S_j after

[1] By *classic* we mean the tables as described in the original Hellman paper.

the column where Y_1 occurs. That is a false alarm, which requires $(t-s)$ encryptions to be detected.

Hellman [100] points out that the expected computation due to false alarms increases the expected computation by at most 50 percent. This reasoning relies on the fact that, for any i, $f^i(Y_1)$ is computed by iterating f i times. However $f^i(Y_1)$ should be computed from Y_i because $f^i(Y_1) = f(Y_i)$. In this case, the computation time required to reach a chain's end is significantly reduced on average while the computation time required to rule out false alarms stays the same. Therefore, false alarms can increase by more than 50% compared to the expected computation. For example, formulas given in Section 10.3 allow determining the computation wasted during the recovery of Windows passwords as proposed in [137]: false alarms increase by 125% compared to the expected computation.

10.2.2 Ruling Out False Alarms Using Checkpoints

Our idea consists in defining a set of positions α_i in the chains to be checkpoints. We calculate the value of a given function G for each checkpoint of each chain j and store these $G(X_{j,\alpha_i})$ with the end of each chain $X_{j,t}$. During the on-line phase, when we generate $Y_1, Y_2, \ldots, Y_s$, we also calculate the values for G at each checkpoint, yielding the values $G(Y_{\alpha_i+s-t})$. If Y_s matches the end of a chain that we have stored, we compare the values of G for each checkpoint that the chain Y has gone through with the values stored in the table. If they differ for at least one checkpoint we know for certain that this is a false alarm. If they are identical, we cannot determine if this is a false alarm without regenerating the chain. In other words, the set of checkpoints behaves like a Monte Carlo test: if the test affirms that a matching end leads to a false alarm, then it is definitely a false alarm: the chain starting from S_j does not need to be regenerated. Sometimes, the test does not answer: we cannot conclude in this case whether or not the matching end will lead to a false alarm: the chain must be regenerated.

In order to be efficient, G should be easily computable and the storage of its output should require few bits. Below, we consider the function G such that $G(X)$ simply outputs the less significant bit of X. Thus we have:

$$\Pr\{G(X_{j,\alpha}) \neq G(Y_{\alpha+s-t}) \mid X_{j,\alpha} \neq Y_{\alpha+s-t}\} = \frac{1}{2}\left(1 - \frac{1}{2^{|K|}}\right) \approx \frac{1}{2}.$$

The case $X_{j,\alpha} \neq Y_{\alpha+s-t}$ occurs when the merge appears after the column α (Figure 10.1). The case $X_{j,\alpha} = Y_{\alpha+s-t}$ occurs when either K appears in the regenerated chain or the merge occurs before the column α (Figure 10.2).

In the next section, we perform an indepth analysis of the rainbow tables' performance. By performance we mean the exact amount of work that the trade-off requires, given the available amount of memory, the probability of success and other parameters. Then, we introduce the checkpoint concept in rainbow tables and analyze both theoretical and practical results.

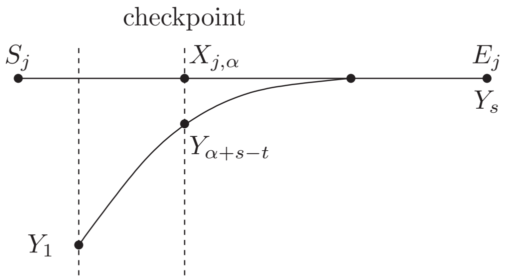


Figure 10.1: False alarm detected with probability 1/2

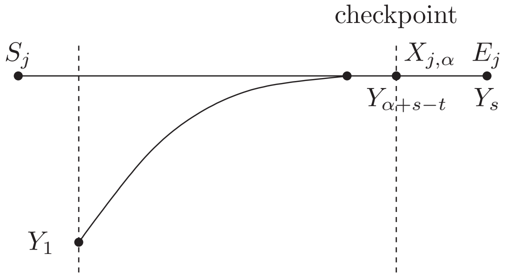


Figure 10.2: False alarm not detected

10.3 Checkpoints Within Perfect Rainbow Tables

10.3.1 Perfect Tables

The key to an efficient trade-off is to ensure that the available memory is used most efficiently. Thus we want to avoid the use of memory to store chains that contain elements that are already part of other chains. To do so, we first generate more chains than we actually need. Then we search for merges and remove chains until there are no merges. The resulting tables are called perfect tables. They have been introduced by Borst, Preneel, and Vandewalle [47] and analyzed by Standaert, Rouvroy, Quisquater, and Legat [163]. Creating perfect rainbow and tables using distinguished points (DP) is easy since merging chains can be recognized by their identical end points. Since end points need to be sorted to facilitate look-ups, identifying the merges comes for free. Classic chains do not have this advantage. Every single element of every classic chain that is generated has to be looked up in all elements of all chains of the same table. This requires $mt\ell$ look-ups in total where ℓ is the number of stored tables.

Perfect classic and DP tables are made of unique elements. In perfect rainbow tables, no element appears twice in any given column, but it may appear more than once across different columns. In all variants of the trade-off, there is a limit to the size of the perfect tables that can be generated. The brute-force way of finding the maximum number of chains of given length t that will not merge is to generate a chain from each of the N possible keys and remove the merges.

In the following sections, we only consider perfect tables.

10.3.2 Optimal Configuration

From [137], we know that the success rate of a single un-perfect rainbow table is

$$1 - \prod_{i=1}^{t} \left(1 - \frac{m_i}{N}\right)$$

where m_i is the number of different keys in column i. With perfect rainbow tables, we have $m_i = m$ for all i such that $1 \leq i \leq t$. The success rate of a single perfect rainbow table is therefore

$$P_{\text{rainbow}} = 1 - \left(1 - \frac{m}{N}\right)^t. \tag{10.1}$$

The fastest cryptanalysis time is reached by using the largest possible perfect tables. This reduces the amount of duplicate information stored in the table and reduces the number of tables that have to be searched. For a given chain length t, the maximum number $m_{\max}(t)$ of rainbow chains that can be generated without merges is obtained (see [137]) by calculating the number of independent elements at column t if we start with N elements in the first column. Thus we have

$$m_{\max}(t) = m_t \quad \text{where} \quad m_1 = N \quad \text{and} \quad m_{n+1} = N\left(1 - e^{-\frac{m_n}{N}}\right) \text{ with } 0 < n < t.$$

For non small t we can find a closed form for $m_{\max}$ by approximating the recurrence relation

$$m_{n+1} = N\left(1 - e^{-\frac{m_n}{N}}\right).$$

Using the Taylor approximation of the exponential we get

$$m_{n+1} \approx N\left(\frac{m_n}{N} - \frac{m_n^2}{2N^2}\right) = m_n - \frac{m_n^2}{2N}$$

which is accurate for small m or non small n. We can transform this expression into a differential equation

$$\frac{dm_n}{dn} = -\frac{m_n^2}{2N},$$

whose solution is

$$m_n = \frac{2N}{n+c}.$$

We get the maximum number of chains of length t by starting with m_0 equal to N and looking for m_t. When m_0 is N we get $c = 2$ thus we find that

$$m_{\max}(t) \approx \frac{2N}{t+2}. \tag{10.2}$$

From Equation 10.1 and Equation 10.2, we deduce the probability of success of a single perfect rainbow table having $m_{\max}$ chains:

$$P_{\text{rainbow}}^{\max} = 1 - \left(1 - \frac{m_{\max}}{N}\right)^t \approx 1 - e^{-t\frac{m_{\max}}{N}} \approx 1 - e^{-2} \approx 86\%. \tag{10.3}$$

Interestingly, for any N and for t not small, this probability tends toward a constant value. Thus the smallest number of tables needed for a trade-off depends only on the desired success rate P. This makes the selection of optimal parameters very easy. Thus we compute the minimum number of tables ℓ such that the probability P of success of the trade-off is at least:

$$1 - (1 - P_{\text{rainbow}}^{\max})^{\ell} \tag{10.4}$$

From Equation 10.4 and Equation 10.3, we get

$$(1 - P) \leq (1 - P_{\text{rainbow}}^{\max})^{\ell} \approx (e^{-t\frac{m_{\max}}{N}})^{\ell}.$$

Thus

$$\frac{-N}{tm_{\max}} \ln(1 - P) \leq \ell. \tag{10.5}$$

From Equation 10.2 and Equation 10.5, we obtain

$$\ell = \left\lceil \frac{-\ln(1 - P)}{2} \right\rceil$$

and subsequently

$$t = \left\lceil \frac{-N}{M} \ln(1 - P) \right\rceil.$$

10.3.3 Performance of the Trade-Off

Having defined the optimal configuration of the trade-off, we now calculate the exact amount of work required during the on-line phase. The simplicity of rainbow tables makes it possible to include the work due to false alarms both for the average and the worst case.

Cryptanalysis with a set of rainbow tables is done by searching for the key in the last column of each table and then searching sequentially through previous columns of all tables. There are thus a maximum of ℓt searches. We calculate the expectation of the cryptanalysis effort by calculating the probability of success and the amount of work for each search k. When searching for a key at position c of a table, the amount of work to generate a chain that goes to the end of the table is $t - c$. The additional amount of work due to a possible false alarm is c since the chain has to be regenerated from the start to c in order to rule out the false alarm. The probability of success in the search k is:

$$p_k = \frac{m}{N}\left(1 - \frac{m}{N}\right)^{k-1}.$$

We now compute the probability of a false alarm during the search k. When we generate a chain from a given ciphertext and look-up the end of the chain in the table, we can either not find a matching end, find the end of the correct chain or find an end that leads to a false alarm. Thus we can say that the probability of a false alarm is equal to one minus the probability of actually finding the key minus the probability of finding no end point. The probability of not finding an end point is the probability that all points that we generate are not part of the chains that lead to the end points. At column i, these are the m_i chains that we used to build the table. The probability of a false alarm at search k (i.e., in column $c = t - \lfloor \frac{k}{\ell} \rfloor$) is thus

$$q_c = 1 - \frac{m}{N} - \prod_{i=c}^{i=t} \left(1 - \frac{m_i}{N}\right) \tag{10.6}$$

where $c = t - \lfloor \frac{k}{\ell} \rfloor$, $m_t = m$, and $m_{i-1} = -N \ln(1 - \frac{m_i}{N})$. When the tables have exactly the maximum number of chains $m_{\max}$, the probability q_c of false alarms in rainbow tables, when searching from column c, can be rewritten in a compact closed form. We replace the term m_i by $m_{\max}(i)$ in Equation 10.6 and we obtain

$$\begin{aligned}
\prod_{i=c}^{i=t} \left(1 - \frac{m_{\max}(i)}{N}\right) &= \prod_{i=c}^{i=t} \left(1 - \frac{2N}{i+2}\frac{1}{N}\right) \\
&= \prod_{i=c}^{i=t} \left(\frac{i}{i+2}\right) \\
&= \frac{c(c+1)}{(t+1)(t+2)}
\end{aligned}$$

that yields

$$q_c = 1 - \frac{m}{N} - \frac{c(c+1)}{(t+1)(t+2)}.$$

The average cryptanalysis time is thus:

$$T = \sum_{\substack{k=1 \\ c=t-\lfloor \frac{k}{\ell} \rfloor}}^{k=\ell t} p_k \left(W(t-c-1) + Q(c)\right) \ell + \left(1 - \frac{m}{N}\right)^{\ell t} \left(W(t) + Q(1)\right) \ell \tag{10.7}$$

where

$$W(x) = \sum_{i=1}^{i=x} i \quad \text{and} \quad Q(x) = \sum_{i=x}^{i=t} q_i i.$$

The second term of Equation 10.7 is the work that is being carried out every time no key is found in the table while the first term corresponds to the work that is being carried out during the search k. W represents the work needed to generate a chain until matching an end point. Q represents the work to rule out a false alarm. We can rewrite Equation 10.7 as follows:

$$\begin{aligned}
T &= \sum_{\substack{k=1 \\ c=t-\lfloor \frac{k}{\ell} \rfloor}}^{k=\ell t} p_k \left(\sum_{i=1}^{i=t-c-1} i + \sum_{i=c}^{i=t} q_i i\right) \ell + (1 - \frac{m}{N})^{\ell t} \left(\sum_{i=1}^{i=t} i + \sum_{i=1}^{i=t} q_i i\right) \ell \\
&= \sum_{\substack{k=1 \\ c=t-\lfloor \frac{k}{\ell} \rfloor}}^{k=\ell t} p_k \left(\frac{1}{2}(t-c)(t-c-1) + \sum_{i=c}^{i=t} q_i i\right) \ell + (1 - \frac{m}{N})^{\ell t} \left(\frac{t(t-1)}{2} + \sum_{i=1}^{i=t} q_i i\right) \ell
\end{aligned}$$

In order to illustrate T, we have run a few experiments aiming at cracking Windows LM Hash passwords [137]. The results are given in Table 10.1. We recall that N is the size of the input space of E, t is the length of the chains, m is the number of chains per table, and ℓ is the number of tables.

$N = 8.06 \times 10^{10}$, $t = 10000$, $m = 15408697$, and $\ell = 4$	theory	measured over 1000 experiments
encryptions (average)	1.55×10^7	1.66×10^7
encryptions (worst case)	2.97×10^8	2.96×10^8
number of false alarms (average)	1140	1233
number of false alarms (worst case)	26048	26026

Table 10.1: Calculated and measured performance of rainbow tables

10.3.4 Checkpoints in Rainbow Tables

From results of Section 10.3.3, we establish below the gain brought by the checkpoints. We firstly consider only one checkpoint α. Let $Y_1 \dots Y_s$ be a chain generated from a given ciphertext C. From Equation 10.6, we know that the probability that $Y_1 \dots Y_s$ merges with a stored chain is q_{t-s}. The expected work due to a false alarm is therefore $q_{t-s}(t-s)$.

We now compute the probability to detect the false alarm thanks to the checkpoint. If the merge occurs before the checkpoint (Figure 10.2) then the false alarm cannot be detected. If the chain is long enough, i.e., $\alpha + s > t$, the merge occurs after the checkpoint (Figure 10.1) with probability q_α. In this case, the false alarm is detected with probability

$$\Pr\left(G(X_{j,\alpha}) \neq G(Y_{\alpha+s-t}) \mid X_{j,\alpha} \neq Y_{\alpha+s-t}\right).$$

We define $g_\alpha(s)$ as follows:

$$g_\alpha(s) = \begin{cases} 0 \text{ if there is no checkpoint in column } \alpha, \\ 0 \text{ if } (\alpha + s) \leq t, \text{ i.e., the chain generated from } Y_1 \text{ does not reach column } \alpha, \\ \Pr\left(G(X_{j,\alpha}) \neq G(Y_{\alpha+s-t}) \mid X_{j,\alpha} \neq Y_{\alpha+s-t}\right) \text{ otherwise.} \end{cases}$$

We rewrite $Q(x)$ as follows

$$Q(x) = \sum_{i=x}^{i=t} i\left(q_i - q_\alpha \cdot g_\alpha(t-i)\right).$$

We now define the *time gain*. Let M, T, N and M', T', N' be the parameters of two trade-offs respectively. Let σ_T such that

$$T' = \sigma_T \cdot T.$$

The time gain of the second trade-off over the first one is simply defined by

$$(1 - \sigma_T) = 1 - \frac{T'}{T}.$$

We apply our checkpoint technique with $N = 8.06 \times 10^{10}$, $t = 10000$, $m = 15408697$, $\ell = 4$ and G as defined in Section 10.2.2. Both theoretical and experimental time gains of the checkpoint-based trade-off over the trade-off without checkpoints, are plotted in Figure 10.3.

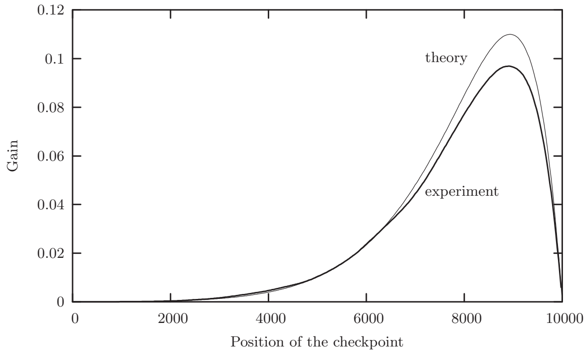


Figure 10.3: Theoretical and experimental time gains when one checkpoint is used

We can now generalize to t checkpoints. We rewrite $Q(x)$ as follows:

$$Q(x) = \sum_{i=x}^{i=t} i \left(q_i - q_i \cdot g_i(t-i) - \sum_{j=i+1}^{j=t} \left(q_j \cdot g_j(t-j) \prod_{k=i}^{k=j-1} (1 - g_k(t-k)) \right) \right).$$

Note that when the time gain is evaluated between two trade-offs, the required memory should be the same in both cases in order to supply a fair comparison. However, storing some checkpoints requires additional memory compared to the trade-off without checkpoints. This amount of memory could be instead used to store more chains. We therefore introduce the concept of *memory cost*. Let σ_M such that

$$M' = \sigma_M \cdot M.$$

The memory cost of a trade-off of parameters M, T, N over another trade-off of parameters M', T', N' is

$$(\sigma_M - 1) = \frac{M'}{M} - 1.$$

Given that $T \propto N^2/M^2$ the time gain becomes:

$$(1 - \frac{T'}{T}) = 1 - \frac{1}{\sigma_M^2}.$$

Thus, given a memory cost, we can compare the time gains when the additional memory is used to store chains and when it is used to store checkpoints. Numerical results are given in Table 10.2: an additional 0.89% of memory saves about 10.99% of cryptanalysis time. This is six times more than the 1.76% of gain that would be obtained by using the same amount of memory to store additional chains. Our checkpoints thus perform much better than the basic trade-off. As we add more and more checkpoints, the gain per checkpoint decreases. In our example it is well worth to use 6 bits of checkpoint values (5.35% of additional memory) per chain to obtain a gain of 32.04%.

Clearly, the results depend on the considered problem, i.e., the function E, and the parameters and implementation of the trade-off. Here, the 0.89% of memory per checkpoint is calculated by assuming that the start and end of the chains are stored in 56 bits each, as

Number of checkpoints	1	2	3	4	5	6
Cost (memory)	0.89%	1.78%	2.67%	3.57%	4.46%	5.35%
Gain if chains (time)	1.76%	3.47%	5.14%	6.77%	8.36%	9.91%
Gain if checkpoints (time)	10.99%	18.03%	23.01%	26.76%	29.70%	32.04%
Optimal checkpoints	8935 ± 5	8565 9220 ± 5	8265 8915 9370 ± 5	8015 8655 9115 9470 ± 5	7800 8450 8900 9250 9550 ± 50	7600 8200 8700 9000 9300 9600 ± 100

Table 10.2: Cost and gain of using checkpoints in password cracking

our example relies on Windows LM Hash that uses DES. As indicated in Section 10.4, the number of bits used to store one chain can be optimized and reduced to 49 bits instead of 112 bits. In this case a bit of checkpoint data adds 2% of memory and it is still well worth using three checkpoints of one bit each to save 23% of work.

10.4 Implementation Tips

For the sake of completeness we want to include a few short remarks on the optimized implementation of the trade-offs. Indeed, an optimized implementation can yield performance gains almost as important as the algorithmic optimizations. We limit our tips to the implementation of rainbow tables.

10.4.1 Storing the Chain's End Points

The number of operations of the trade-off decreases by the square of the available memory. Since available memory is measured in bytes and not in number of chains, it is important to choose an efficient format for storing the chains. A first issue is whether to use inputs or outputs of the function to be inverted (keys or ciphertexts) as beginning and end of chains. In practice, keys are commonly smaller than ciphertexts. It is thus more efficient to store keys (the key at the end of the chain has no real function but the extra reduction needed to generate it from the last ciphertext is well worth the saved memory). A second and more important issue is taking advantage of the way the tables are organized. Indeed a table consists of pairs of beginnings and ends of chains. To facilitate look-ups the chains are sorted by increasing values of the chain ends. Since the ends are sorted, successive ends often have an identical prefix. As suggested in [40] we can thus remove a certain length of prefix and replace it by an index table that indicates where every prefix starts in the table.

In our Windows password example, there are about 2^{37} keys of 56 bits. Instead of storing the 56 bits, we store a 37 bit index. From this index we take 21 bits as prefix and store only the last 16 bits in memory. We also store a table with 2^{21} entries that point to the corresponding suffixes for each possible prefix.

10.4.2 Storing the Chain's Starting Points

The set of keys used for generating all the chains is usually smaller that the total set of keys. Since rainbow tables allow choosing the starting points at will, we can use keys with increasing value of their index. In our example we used about 300 million starting points. This value can be expressed in 29 bits, so we only need to store the 29 lower bits of the index. The total amount of memory needed to store a chain is thus $29 + 16$ bits for the start and the end. The table that relates the prefixes to the suffixes incurs about 3.5 bits per chain. Thus, altogether we need 49 bits per chain. A simple implementation that stores all 56 bits of the start and end chain would need 2.25 times more memory and be 5 times slower.

10.4.3 Storing the Checkpoints

For reasons of memory access efficiency it may in some implementations be more efficient to store the start and the end of a chain (that is, its suffix) in multiples of 8 bits. If the size of some parameters does not exactly match the size of the memory units, the spare bits can be used to store checkpoints for free. In our case, the 29 bits of the chain's starting point are stored in a 32 bit word, leaving 3 bits available for checkpoints.

10.5 Further Research

Besides having better performance, checkpoints can be generated almost for free while generating the trade-off tables. Thus, there is no indication for not using checkpoints and we conjecture that they will be used in many future implementations of trade-offs. Also, checkpoints are a new concept in time-memory trade-offs and they may lead to other optimizations and applications. Below we give a new approach that has been put forward by Finiasz [77].

The basic idea consists in using the checkpoints during the precomputation phase in order to select the chains that will be stored in the tables. Assume that two chains merge during the precomputation phase such that the collision appears in column c. If there exist a checkpoint α such that $\alpha \leq c$ and the parity bits of the values of the chains in column c are different, both chains are stored. Thus, two starts of chain and only one end of chain are required in order to store two chains. Intuitively, the merge should appear at about the end of the chains, otherwise this technique would not be efficient.

During the online phase, only one of these two chains is regenerated when a (false) alarm occurs, according to the parity bit of the checkpoint. We could still generalize this technique if the checkpoint is no longer one bit parity but a k-bit check. Thus, 2^k chains could be stored using only one end of chain. Clearly, this generalization will be rapidly limited by the difficulty to generate such merging chains during the precomputation phase.

We will deal with this new approach in the near future.

Conclusion

Fair exchange has been intensively studied during past twenty years. In the protocols first proposed, item exchange was either done in a gradual way to reduce losses in the event of a dishonest participant, or with the assistance of a centralized trusted third party who was involved in each exchange. With the introduction of optimistic protocols in the mid-nineties, fairness rested on a centralized trusted third party who was involved in the exchange only in the event of a conflict between participants. While a majority of the recent works is based on this model, in this thesis we have proposed two alternative approaches.

The first approach that we proposed consists of attaching to each participant a *guardian angel*, that is, a security module conceived by a trustworthy authority and whose behavior cannot deviate from the established rules. Thus, the guardian angels exchange the expected items through a secure channel and execute a protocol, *Keep-in-Touch* (KiT), whose goal is to safely close the exchange. The fair exchange protocol fails if and only if the KiT protocol fails, which arises only if the last message exchanged between the guardian angels is dropped. Since the number of expected exchanged messages in the KiT is randomly chosen and unknown by the adversary, the only way to violate fairness is to drop a message randomly. Hence the probability of unfairness can be made as low as desired if one increases the number of exchanges during the KiT. It is a new approach that does not rest on a centralized trusted third party and is also not gradual. We then used results from the distributed algorithms' consensus problem to generalize this approach to the multi-party fair exchange. In doing so, we have introduced the *Gracefully Degrading Fair Exchange* (GDFE). The idea is that fairness is ensured in a deterministic way if there are a majority of honest participants, but is ensured only in a probabilistic way in the contrary case. In this case, the probability that the protocol fails can be made as low as desired if one increases the number of exchanges between the participants, as we saw in the two-party case. The second approach too does not use a centralized trusted third party but here, fairness lies on the participant's neighbors. Indeed, fair exchanges are generally not carried out in a closed environment, but in an environment where other parties are present who, *a priori*, do not participate in the exchange. The idea is to require them to restore fairness if (and only if) a conflict occurs between the two participants.

In the second part of this thesis, our research was directed towards Radio Frequency Identification (RFID). In particular, we studied the *traceability* problems in RFID systems. Since this domain is a recent one, works that were published until now are generally quite informal. We have provided an overview of the RFID technology, and made clear the systems on which research focuses itself at this moment. Even though current works speaks of "RFID protocols", we have shown that it is necessary to distinguish between protocols whose goals are identification and those whose goals are authentication. We then analyzed the risks in terms of security and privacy. In particular, we showed the link between the communication model and traceability. Later, we analyzed several protocols and presented their weaknesses. We were also interested in the complexity issues and proposed a technique based on time-memory trade-off, whose goal is to reduce the tags' identification time.

RFID is only at its first development stages and today, several research directions have opened up. In the short run, the main problem that will have to be dealt with is the formalization of the adversary model. This fundamental aspect must be discussed in order to move ahead in protocol analysis. Today, too many protocols have failed due to an unrealistic adversary model. One would probably have to distinguish two cases: on the one hand protocols that only require very few resources, therefore intended for very low cost tags, and thus cannot guarantee strong security. Their objective would be to counter attacks that are easy to carry out, for examples passive attacks. And on the other hand, protocols intended for tags able to use symmetric cryptography and consequently be able to guarantee a satisfactory security. We have seen that a complexity problem surfaces in this case. An open question today is, whether it is possible to conceive identification protocols resistant to malicious traceability with a complexity better than a linear complexity. An equally interesting question is to know whether it is possible to wholly identify a system with a complexity better than a quadratic complexity. We also think that the relay-attack risks are far from negligible, notably in access control applications. This problem is amplified by the fact that tags answer without their carrier's agreement. A reflection on the distance bounding protocols adapted to RFID will have to be made. Another interesting issue is the ownership transfer of tags. This problem that has been tackled by Molnar, Soppera, and Wagner [134, 135] deserves serious consideration.

Lastly, our works on RFID led us to study more thoroughly the time-memory trade-offs. In this framework, we proposed a technique using *checkpoints* that allows detecting false alarms in a probabilistic manner, and consequently reduces the cryptanalysis time. The checkpoints show the possibilities of other new prospects in the time-memory trade-offs domain. A possible future work could be using checkpoints in the precomputation phase rather than in the cryptanalysis phase.

Bibliography

[1] Priya Agrawal, Neha Bhargava, Chaitra Chandrasekhar, Al Dahya, and J.D. Zamfirescu. The MIT ID Card System: Analysis and recommendations, December 2004. MIT, Massachusetts, USA. (68)

[2] Manfred Aigner and Martin Feldhofer. Secure symmetric authentication for RFID tags. In *Telecommunication and Mobile Computing – TCMC 2005*, Graz, Austria, March 2005. (105)

[3] Katherine Albrecht and Liz McIntyre. *Spychips: How Major Corporations and Government Plan to Track Your Every Move with RFID.* Nelson Current, 2005. (70)

[4] Ross Anderson and Markus Kuhn. Low cost attacks on tamper resistant devices. In Bruce Christianson, Bruno Crispo, Mark Lomas, and Michael Roe, editors, *International Workshop on Security Protocols – IWSP'97*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136, Paris, France, April 1997. Springer-Verlag. (63)

[5] N. Asokan. *Fairness in Electronic Commerce.* PhD thesis, University of Waterloo, Waterloo, Canada, May 1998. (7, 8)

[6] N. Asokan, Birgit Baum-Waidner, Matthias Schunter, and Michael Waidner. Optimistic synchronous multi-party contract signing. Research Report RZ 3089, IBM Research Division, Zurich, Switzerland, December 1998. (11)

[7] N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. Research Report RZ 2858, IBM Research Division, Zurich, Switzerland, September 1996. (8, 14)

[8] N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for multi-party fair exchange. Research Report RZ 2892, IBM Research Division, Zurich, Switzerland, December 1996. (11)

[9] N. Asokan, Matthias Schunter, and Michael Waidner. Optimistic protocols for fair exchange. In *Conference on Computer and Communications Security – CCS'97*, pages 7–17, Zurich, Switzerland, April 1997. ACM, ACM Press. (8, 13, 14)

[10] N. Asokan, Victor Shoup, and Michael Waidner. Asynchronous protocols for optimistic fair exchange. In *IEEE Symposium on Research in Security and Privacy*, pages 86–99, Oakland, California, USA, May 1998. IEEE, IEEE Computer Society Press. (8, 9, 14)

[11] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic fair exchange of digital signatures. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606, Helsinki, Finland, May–June 1998. IACR, Springer-Verlag. (8)

[12] Giuseppe Ateniese, Jan Camenisch, and Breno de Medeiros. Untraceable RFID tags via insubvertible encryption. In *Conference on Computer and Communications Security – CCS'05*, Alexandria, Virginia, USA, November 2005. ACM, ACM Press. (89)

[13] Gildas Avoine. Security and privacy in RFID systems. Online bibliography available at http://lasecwww.epfl.ch/∼gavoine/rfid/. (4)

[14] Gildas Avoine. Introduction aux protocoles d'échange équitable (Introduction to fair exchange). *Multi-System & Internet Security Cookbook – MISC*, (4):10–11, November 2002. (2)

[15] Gildas Avoine. Privacy issues in RFID banknote protection schemes. In Jean-Jacques Quisquater, Pierre Paradinas, Yves Deswarte, and Anas Abou El Kadam, editors, *International Conference on Smart Card Research and Advanced Applications – CARDIS*, pages 33–48, Toulouse, France, August 2004. IFIP, Kluwer Academic Publishers. (1, 4, 70, 81, 94)

[16] Gildas Avoine. Adversary model for radio frequency identification. Technical Report LASEC-REPORT-2005-001, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, September 2005. (1, 4, 77)

[17] Gildas Avoine. Fraud within asymmetric multi-hop cellular networks. In Andrew Patrick and Moti Yung, editors, *Financial Cryptography – FC'05*, volume 3570 of *Lecture Notes in Computer Science*, pages 1–15, Roseau, The Commonwealth Of Dominica, February–March 2005. IFCA, Springer-Verlag. (4, 29)

[18] Gildas Avoine. Scalability issues in RFID systems. Ecrypt Workshop on RFID and Lightweight Crypto (Invited talk), July 2005. (4)

[19] Gildas Avoine, Etienne Dysli, and Philippe Oechslin. Reducing time complexity in RFID systems. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography – SAC 2005*, Lecture Notes in Computer Science, Kingston, Canada, August 2005. Springer-Verlag. (1, 4, 70, 110, 115, 125)

[20] Gildas Avoine, Felix Gärtner, Rachid Guerraoui, and Marko Vukolić. Gracefully degrading fair exchange with security modules. Technical Report EPFL-I&C 26, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, March 2004. (1, 2, 31)

[21] Gildas Avoine, Felix Gärtner, Rachid Guerraoui, and Marko Vukolić. Modern security with traditional distributed algorithms. Ecrypt Workshop on Secure Multi-Party Computation, October 2004. (2)

[22] Gildas Avoine, Felix Gärtner, Rachid Guerraoui, and Marko Vukolić. Gracefully degrading fair exchange with security modules. In Mario Dal Cin, Mohamed Kaâniche, and András Pataricza, editors, *European Dependable Computing Conference – EDCC-5*, volume 3463 of *Lecture Notes in Computer Science*, pages 55–71, Budapest, Hungary, April 2005. Springer-Verlag. (1, 2, 31)

[23] Gildas Avoine, Pascal Junod, and Philippe Oechslin. Time-memory trade-offs: False alarm detection using checkpoints. In *Progress in Cryptology – Indocrypt 2005*, volume 3797 of *Lecture Notes in Computer Science*, pages 183–196, Bangalore, India, December 2005. Cryptology Research Society of India, Springer-Verlag. (1, 4, 125, 131)

[24] Gildas Avoine, Pascal Junod, and Philippe Oechslin. Time-memory trade-offs: False alarm detection using checkpoints. Technical Report LASEC-REPORT-2005-002, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, September 2005. (1, 4, 131)

[25] Gildas Avoine, Jean Monnerat, and Thomas Peyrin. Advances in alternative non-adjacent form representations. In Anne Canteaut and Kapaleeswaran Viswanathan, editors, *Progress in Cryptology – Indocrypt 2004*, volume 3348 of *Lecture Notes in Computer Science*, pages 260–274, Chennai, India, December 2004. Cryptology Research Society of India, Springer-Verlag. (4)

[26] Gildas Avoine and Philippe Oechslin. RFID traceability: A multilayer problem. In Andrew Patrick and Moti Yung, editors, *Financial Cryptography – FC'05*, volume 3570 of *Lecture Notes in Computer Science*, pages 125–140, Roseau, The Commonwealth Of Dominica, February–March 2005. IFCA, Springer-Verlag. (1, 4, 70, 78)

[27] Gildas Avoine and Philippe Oechslin. A scalable and provably secure hash based RFID protocol. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pages 110–114, Kauai Island, Hawaii, USA, March 2005. IEEE, IEEE Computer Society Press. (1, 4, 115, 125)

[28] Gildas Avoine and Serge Vaudenay. Cryptography with guardian angels: Bringing civilization to pirates. *Report on a Working Session on Security in Wireless Ad Hoc Networks, Levente Buttyán and Jean-Pierre Hubaux editors, ACM Mobile Computing and Communications Review*, 7(1):74–94, January 2003. (1, 2, 19)

[29] Gildas Avoine and Serge Vaudenay. Fair exchange with guardian angels. In Kijoon Chae and Moti Yung, editors, *International Workshop on Information Security Applications – WISA 2003*, volume 2908 of *Lecture Notes in Computer Science*, pages 188–202, Jeju Island, Korea, August 2003. Springer-Verlag. (1, 2, 19)

[30] Gildas Avoine and Serge Vaudenay. Optimistic fair exchange based on publicly verifiable secret sharing. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Australasian Conference on Information Security and Privacy – ACISP'04*, volume 3108 of *Lecture Notes in Computer Science*, pages 74–85, Sydney, Australia, July 2004. Springer-Verlag. (1, 2, 45)

[31] Alireza Bahreman and Doug Tygar. Certified electronic mail. In *Symposium on Network and Distributed System Security – NDSS'94*, pages 3–19. Internet Society, February 1994. (8)

[32] Feng Bao, Robert Deng, and Wenbo Mao. Efficient and practical fair exchange protocols with off-line TTP. In *IEEE Symposium on Research in Security and Privacy*, pages 77–85, Oakland, California, USA, May 1998. IEEE, IEEE Computer Society Press. (9)

[33] Feng Bao, Robert Deng, Khanh Quoc Nguyen, and Vijay Varadharajan. Multi-party fair exchange with an off-line trusted neutral party. In *International Workshop on Databases and Expert Systems Applications – DEXA*, pages 858–862, Florence, Italy, September 1999. IEEE. (11)

[34] Michael Ben-Or, Oded Goldreich, Silvio Micali, and Ronald Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, January 1990. (8)

[35] Zinaida Benenson, Felix Gärtner, and Dogan Kesdogan. Secure multi-party computation with security modules. In Hannes Federrath, editor, *Sicherheit 2005*, volume 62 of *Lecture Notes in Informatics*, pages 41–52, Regensburg, Germany, April 2005. Gesellschaft für Informatik (GI). (33)

[36] Elwyn Berlekamp, Robert McEliece, and Henk van Tilborg. On the inherent intractability of certain coding problems. *IEEE Transactions on Information Theory*, 24(3):384–386, May 1978. (111)

[37] Philip Bernstein, Vassos Hadzilacos, and Nathan Goodman. *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, Boston, Massachusetts, USA, 1987. (32)

[38] Thomas Beth and Yvo Desmedt. Identification tokens – or: Solving the chess grandmaster problem. In Alfred Menezes and Scott Vanstone, editors, *Advances in Cryptology – CRYPTO'90*, volume 537 of *Lecture Notes in Computer Science*, pages 169–176, Santa Barbara, California, USA, August 1990. IACR, Springer-Verlag. (71)

[39] Bibliotheca RFID Library Systems AG. `http://www.bibliotheca-rfid.com`. (58)

[40] Alex Biryukov, Adi Shamir, and David Wagner. Real time cryptanalysis of A5/1 on a PC. In Bruce Schneier, editor, *Fast Software Encryption – FSE'00*, volume 1978 of *Lecture Notes in Computer Science*, pages 1–18, New York, USA, April 2000. Springer-Verlag. (141)

[41] George Robert Blakley. Safeguarding cryptographic keys. In Richard Merwin, Jacqueline Zanca, and Merlin Smith, editors, *National Computer Conference*, volume 48 of *American Federation of Information Processing Societies*, pages 313–317, New York, USA, June 1979. AFIPS Press. (45, 47)

[42] Avrim Blum, Merrick Furst, Michael Kearns, and Richard Lipton. Cryptographic primitives based on hard learning problems. In Yvo Desmedt, editor, *Advances in Cryptology – CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 278–291, Santa Barbara, California, USA, August 1994. IACR, Springer-Verlag. (111)

[43] Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the Association for Computing Machinery*, 50(4):506–519, July 2003. (111)

[44] Manuel Blum. How to exchange (secret) keys. *ACM Transactions on Computer Systems*, 1(2):175–193, May 1983. (8)

[45] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT'01*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532, Gold Coast, Australia, December 2001. IACR, Springer-Verlag. (98)

[46] Dan Boneh and Moni Naor. Timed commitments. In Mihir Bellare, editor, *Advances in Cryptology – CRYPTO'00*, volume 1880 of *Lecture Notes in Computer Science*, pages 236–254, Santa Barbara, California, USA, August 2000. IACR, Springer-Verlag. (8)

[47] Johan Borst, Bart Preneel, and Joos Vandewalle. On the time-memory tradeoff between exhaustive key search and table precomputation. In Peter de With and Mihaela van der Schaar-Mitrea, editors, *Symposium on Information Theory in the Benelux*, pages 111–118, Veldhoven, The Netherlands, May 1998. (132, 135)

[48] Stefan Brands and David Chaum. Distance-bounding protocols. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 344–359, Lofthus, Norway, May 1993. IACR, Springer-Verlag. (71)

[49] Ernest Brickell, David Chaum, Ivan Damgård, and Jeroen van de Graaf. Gradual and verifiable release of a secret. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*, pages 156–166, Santa Barbara, California, USA, August 1988. IACR, Springer-Verlag. (8)

[50] Holger Bürk and Andreas Pfitzmann. Value exchange systems enabling security and unobservability. *Computers and Security*, 9(8):715–721, December 1990. (9)

[51] Laurent Bussard and Walid Bagga. Distance-bounding proof of knowledge to avoid real-time attacks. In *International Information Security Conference – IFIP/SEC 2005*, Makuhari-Messe, Chiba, Japan, May–June 2005. Kluwer. (71)

[52] Srdjan Capkun and Jean-Pierre Hubaux. Secure positioning in wireless networks. *IEEE Journal on Selected Areas in Communications: Special Issue on Security in Wireless Ad Hoc Networks*, 2006. (71)

[53] Auto-ID Center. Draft protocol specification for a 900 MHz class 0 radio frequency identification tag. `http://www.epcglobalinc.org`, February 2003. EPCGlobal Inc. (82)

[54] David Chaum and Torben Pedersen. Wallet databases with observers. In Ernest Brickell, editor, *Advances in Cryptology – CRYPTO'92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105, Santa Barbara, California, USA, August 1992. IACR, Springer-Verlag. (26)

[55] Benny Chor, Shafi Goldwasser, Silvio Micali, and Baruch Awerbuch. Verifiable secret sharing and achieving simultaneity in the presence of faults. In *IEEE Symposium on Foundations of Computer Science – FOCS'85*, pages 383– 395, Portland, Oregon, USA, October 1985. IEEE. (47)

[56] Richard Cleve. Controlled gradual disclosure schemes for random bits and their applications. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 573–588, Santa Barbara, California, USA, August 1990. IACR, Springer-Verlag. (8)

[57] Tom Coffey and Puneet Saidha. Non-repudiation with mandatory proof of receipt. *ACM Computer Communication Review*, 26, 1996. (8)

[58] Benjamin Cox, Doug Tygar, and Marvin Sirbu. NetBill security and transaction protocol. In *Workshop on Electronic Commerce – EC'95*, pages 77–88, New York, USA, July 1995. USENIX. (8)

[59] Ivan Damgård. Practical and probably secure release of a secret and exchange of signatures. In Tor Helleseth, editor, *Advances in Cryptology – EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 200–217, Lofthus, Norway, May 1993. IACR, Springer-Verlag. (8)

[60] Robert Deng, Li Gong, Aurel Lazar, and Weiguo Wang. Practical protocols for certified electronic mail. *International Journal of Network and Systems Management*, 4(3):279–297, September 1996. (8)

[61] Dorothy Denning. *Cryptography and Data Security*, page 100. Addison-Wesley, Boston, Massachusetts, USA, June 1982. (132)

[62] Yvo Desmedt, Claude Goutier, and Samy Bengio. Special uses and abuses of the fiat-shamir passport protocol. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*, pages 21–39, Santa Barbara, California, USA, August 1988. IACR, Springer-Verlag. (70)

[63] Tim Dierks and Christopher Allen. The TLS protocol – version 1.0, January 1999. IETF – RFC 2246. (20)

[64] Tassos Dimitriou. A lightweight RFID protocol to protect against traceability and cloning attacks. In *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005*, Athens, Greece, September 2005. IEEE. (110)

[65] Sandra Dominikus, Elisabeth Oswald, and Martin Feldhofer. Symmetric authentication for RFID systems in practice. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005. Ecrypt. (105)

[66] Joan Dyer, Mark Lindemann, Ronald Perez, Reiner Sailer, Leendert van Doorn, Sean Smith, and Steve Weingart. Building the IBM 4758 secure coprocessor. *IEEE Computer Society*, 34(10):57–66, October 2001. (27)

[67] Electronic Product Code Global Inc. `http://www.epcglobalinc.org`. (64, 79)

[68] Taher Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, July 1985. (48, 88, 98)

[69] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald Rivest, and Alan Sherman, editors, *Advances in Cryptology – CRYPTO'82*, pages 205–210, Santa Barbara, California, USA, August 1982. Plenum Press. (8)

[70] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, June 1985. (8)

[71] Shimon Even and Yacov Yacobi. Relations amoung public key signature systems. Technical Report 175, Computer Science Department, Technion, Haifa, Israel, 1980. (1, 8, 12, 21)

[72] Benjamin Fabian, Oliver Günther, and Sarah Spiekermann. Security analysis of the object name service for RFID. In *International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing – SecPerU'05*, Santorini Island, Greece, July 2005. IEEE, IEEE Computer Society Press. (70)

[73] Martin Feldhofer, Sandra Dominikus, and Johannes Wolkerstorfer. Strong authentication for RFID systems using the AES algorithm. In Marc Joye and Jean-Jacques Quisquater, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 357–370, Boston, Massachusetts, USA, August 2004. IACR, Springer-Verlag. (70, 77, 105, 107, 113)

[74] Amos Fiat and Moni Naor. Rigorous time/space tradeoffs for inverting functions. In *ACM Symposium on Theory of Computing – STOC'91*, pages 534–541, New Orleans, Louisiana, USA, May 1991. ACM, ACM Press. (132)

[75] Amos Fiat and Moni Naor. Rigorous time/space tradeoffs for inverting functions. *SIAM Journal on Computing*, 29(3):790–803, December 1999. (132)

[76] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, California, USA, August 1986. IACR, Springer-Verlag. (70)

[77] Matthieu Finiasz. Personal communication, May 2005. (142)

[78] Klaus Finkenzeller. *RFID Handbook*. Wiley, England, second edition, 2003. (70)

[79] Michael Fischer, Nancy Lynch, and Michael Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the Association for Computing Machinery*, 32(2):374–382, April 1985. (32)

[80] Kenneth Fishkin, Sumit Roy, and Bing Jiang. Some methods for privacy in RFID communication. In Claude Castelluccia, Hannes Hartenstein, Christof Paar, and Dirk Westhoff, editors, *European Workshop on Security in Ad-hoc and Sensor Networks – ESAS'04*, volume 3313 of *Lecture Notes in Computer Science*, pages 42–53, Heidelberg, Germany, August 2005. Springer-Verlag. (72)

[81] Matt Franklin and Michael Reiter. Fair exchange with a semi-trusted third party. In *Conference on Computer and Communications Security – CCS'97*, pages 1–5, Zurich, Switzerland, April 1997. ACM, ACM Press. (8, 13, 19)

[82] Matt Franklin and Gene Tsudik. Secure group barter: Multi-party fair exchange with semi-trusted neutral parties. In Rafael Hirschfeld, editor, *Financial Cryptography – FC'98*, volume 1465 of *Lecture Notes in Computer Science*, pages 90–102, Anguilla, British West Indies, February 1998. IFCA, Springer-Verlag. (11)

[83] Eiichiro Fujisaki and Tatsuaki Okamoto. A practical and provably secure scheme for publicly verifiable secret sharing and its applications. In Kaisa Nyberg, editor, *Advances in Cryptology – EUROCRYPT'98*, volume 1403 of *Lecture Notes in Computer Science*, pages 32–46, Helsinki, Finland, May–June 1998. IACR, Springer-Verlag. (47)

[84] Eiichiro Fujisaki and Tatsuaki Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 537–554, Santa Barbara, California, USA, August 1999. IACR, Springer-Verlag. (99)

[85] Juan Garay, Markus Jakobsson, and Philip MacKenzie. Abuse-free optimistic contract signing. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466, Santa Barbara, California, USA, August 1999. IACR, Springer-Verlag. (9)

[86] Juan Garay and Philip MacKenzie. Abuse-free multi-party contract signing. In Prasad Jayanti, editor, *Distributed Computing – DISC'99*, volume 1693 of *Lecture Notes in Computer Science*, pages 151–165, Bratislava, Slovak Republic, September 1999. Springer-Verlag. (10)

[87] Simson Garfinkel. Adopting fair information practices to low cost RFID systems. Ubicomp 2002 – Workshop on Socially-informed Design of Privacy-enhancing Solutions in Ubiquitous Computing, September 2002. (75)

[88] Simson Garfinkel. An RFID bill of rights. *Technology Review*, October 2002. (75)

[89] Simson Garfinkel and Beth Rosenberg (Eds). *RFID: Applications, Security, and Privacy.* Addison Wesley Professional, 2005. (70)

[90] Felix Gärtner, Henning Pagnia, and Holger Vogt. Approaching a formal definition of fairness in electronic commerce. In *International Workshop on Electronic Commerce – WELCOM'99*, pages 354–359, Lausanne, Switzerland, October 1999. IEEE, IEEE Computer Society Press. (9)

[91] Henri Gilbert, Matthew Robshaw, and Hervé Sibert. An active attack against HB^+ – a provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237, 2005. IACR. (112)

[92] Marc Girault and David Lefranc. Public key authentication with one (online) single addition. In Marc Joye and Jean-Jacques Quisquater, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2004*, volume 3156 of *Lecture Notes in Computer Science*, pages 413–427, Boston, Massachusetts, USA, August 2004. IACR, Springer-Verlag. (75)

[93] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal re-encryption for mixnets. In Tatsuaki Okamoto, editor, *The Cryptographers' Track at the RSA Conference – CT-RSA*, volume 2964 of *Lecture Notes in Computer Science*, pages 163–178, San Francisco, California, USA, February 2004. Springer-Verlag. (76, 85, 88, 89, 90)

[94] Li Gong. New protocols for third-party-based authentication and secure broadcast. In *Conference on Computer and Communications Security – CCS'94*, pages 184–192, Fairfax, Virginia, USA, April 1994. ACM, ACM Press. (16)

[95] Nicolás González-Deleito and Olivier Markowitch. Exclusion-freeness in multi-party exchange protocols. In Agnes Hui Chan and Virgil Gligor, editors, *Information Security Conference – ISC 2002*, volume 2433 of *Lecture Notes in Computer Science*, pages 200–209, São Paulo, Brazil, September 2002. Springer-Verlag. (10)

[96] Nathan Good, John Han, Elizabeth Miles, David Molnar, Deirdre Mulligan, Laura Quilter, Jennifer Urban, and David Wagner. Radio frequency identification and privacy with information goods. In Sabrina De Capitani di Vimercati and Paul Syverson, editors, *Workshop on Privacy*

in the Electronic Society – WPES, pages 41–42, Washington, DC, USA, October 2004. ACM, ACM Press. (70)

[97] Jim Gray. Notes on data base operating systems. In Rudolph Bayer, Robert Graham, and Gerhard Seegmüller, editors, *Operating Systems, An Advanced Course*, volume 60 of *Lecture Notes in Computer Science*, pages 393–481. Springer-Verlag, 1978. (21)

[98] Gerhard Hancke and Markus Kuhn. An RFID distance bounding protocol. In *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005*, Athens, Greece, September 2005. IEEE. (71)

[99] Johan Håstad. Some optimal inapproximability results. In *ACM Symposium on Theory of Computing – STOC'97*, pages 1–10, El Paso, Texas, USA, May 1997. ACM, ACM Press. (111)

[100] Martin Hellman. A cryptanalytic time-memory trade off. *IEEE Transactions on Information Theory*, IT-26(4):401–406, July 1980. (125, 131, 132, 134)

[101] Dirk Henrici and Paul Müller. Hash-based enhancement of location privacy for radio-frequency identification devices using varying identifiers. In Ravi Sandhu and Roshan Thomas, editors, *International Workshop on Pervasive Computing and Communication Security – PerSec 2004*, pages 149–153, Orlando, Florida, USA, March 2004. IEEE, IEEE Computer Society Press. (85, 86, 87)

[102] Dirk Henrici and Paul Müller. Tackling security and privacy issues in radio frequency identification devices. In Alois Ferscha and Friedemann Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 219–224, Vienna, Austria, April 2004. Springer-Verlag. (76)

[103] Thomas Hjorth. Supporting privacy in RFID systems. Master thesis, Technical University of Denmark, Lyngby, Denmark, December 2004. (70)

[104] Nicholas Hopper and Manuel Blum. Secure human identification protocols. In Colin Boyd, editor, *Advances in Cryptology – ASIACRYPT'01*, volume 2248 of *Lecture Notes in Computer Science*, pages 52–66, Gold Coast, Australia, December 2001. IACR, Springer-Verlag. (111)

[105] International Organization for Standardization. `http://www.iso.org`. (64, 79)

[106] ISO/IEC 18000-1. Information technology AIDC techniques – RFID for item management – air interface, part 1: Generic parameters for air interface communication for globally accepted frequencies. Published standard, September 2004. International Organization for Standardization. (78)

[107] ISO/IEC 18000-3. Information technology AIDC techniques – RFID for item management – air interface – part 3: Parameters for air interface communications at 13.56 MHz. `http://www.iso.org`, February 2003. International Organization for Standardization. (80)

[108] ISO/IEC 7498-1:1994. Information technology – open systems interconnection – basic reference model: The basic model, November 1994. International Organization for Standardization. (78)

[109] Markus Jakobsson. Ripping coins for a fair exchange. In Louis Guillou and Jean-Jacques Quisquater, editors, *Advances in Cryptology – EUROCRYPT'95*, volume 921 of *Lecture Notes in Computer Science*, pages 220–230, Saint Malo, France, May 1995. IACR, Springer-Verlag. (8)

[110] Markus Jakobsson, Jean-Pierre Hubaux, and Levente Buttyán. A micro-payment scheme encouraging collaboration in multi-hop cellular networks. In Rebecca Wright, editor, *Financial Cryptography – FC'03*, volume 2742 of *Lecture Notes in Computer Science*, pages 15–33, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer-Verlag. (29)

[111] Yang Jeongkyu. Security and privacy on authentication protocol for low-cost radio frequency identification. Master thesis, Information and Communications University, Daejeon, Korea, December 2004. (70, 85, 103)

[112] Ari Juels. Minimalist cryptography for low-cost RFID tags. In Carlo Blundo and Stelvio Cimato, editors, *International Conference on Security in Communication Networks – SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 149–164, Amalfi, Italia, September 2004. Springer-Verlag. (63, 70, 76, 77, 85, 92, 93, 94)

[113] Ari Juels. RFID security and privacy: A research survey. *IEEE Journal on Selected Areas in Communication*, 2006. (70)

[114] Ari Juels and John Brainard. Soft blocking: Flexible blocker tags on the cheap. In Sabrina De Capitani di Vimercati and Paul Syverson, editors, *Workshop on Privacy in the Electronic Society – WPES*, pages 1–7, Washington, DC, USA, October 2004. ACM, ACM Press. (74)

[115] Ari Juels, David Molnar, and David Wagner. Security and privacy issues in e-passports. In *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm 2005*, Athens, Greece, September 2005. IEEE. (72)

[116] Ari Juels and Ravikanth Pappu. Squealing euros: Privacy protection in RFID-enabled banknotes. In Rebecca Wright, editor, *Financial Cryptography – FC'03*, volume 2742 of *Lecture Notes in Computer Science*, pages 103–121, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer-Verlag. (74, 76, 81, 85, 94, 95, 97, 99, 102, 103)

[117] Ari Juels, Ronald Rivest, and Michael Szydlo. The blocker tag: Selective blocking of RFID tags for consumer privacy. In Vijay Atluri, editor, *Conference on Computer and Communications Security – CCS'03*, pages 103–111, Washington, DC, USA, October 2003. ACM, ACM Press. (70, 74, 78)

[118] Ari Juels and Stephen Weis. Authenticating pervasive devices with human protocols. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO'05*, volume 3621 of *Lecture Notes in Computer Science*, pages 293–308, Santa Barbara, California, USA, August 2005. IACR, Springer-Verlag. (70, 77, 105, 111, 112, 113)

[119] Günter Karjoth and Paul Moskowitz. Disabling RFID tags with visible confirmation: Clipped tags are silenced. In *Workshop on Privacy in the Electronic Society – WPES*, Alexandria, Virginia, USA, November 2005. ACM, ACM Press. (74)

[120] Jonathan Katz and Ji-Sun Shin. On parallel and concurrent security of HB and HB$^+$. CRYPTO'05, Rump Session, Santa Barbara, California, USA, August 2005. IACR. (112)

[121] Iljun Kim and Tsutomu Matsumoto. Achieving higher success probability in time-memory trade-off cryptanalysis without increasing memory size. *IEICE Transactions on Communications/Electronics/Information and Systems*, E82-A(1):123–, January 1999. (132)

[122] Heiko Knospe and Hartmut Pohl. RFID security. *Information Security Technical Report*, 9(4):39–50, November–December 2004. (70)

[123] Steve Kremer. *Formal Analysis of Optimistic Fair Exchange Protocols*. PhD thesis, Université Libre de Bruxelles, Bruxelles, Belgium, December 2003. (7, 9)

[124] Koji Kusuda and Tsutomu Matsumoto. Optimization of time-memory trade-off cryptanalysis and its application to DES, FEAL-32, and Skipjack. *IEICE Transactions on Fundamentals*, E79-A(1):35–48, January 1996. (132)

[125] Nancy Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, San Francisco, California, USA, 1996. (21, 37, 38)

[126] Jessie MacWilliams and Neil Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 1977. (111)

[127] Olivier Markowitch. *Les protocoles de non-répudiation.* PhD thesis, Université Libre de Bruxelles, Bruxelles, Belgium, January 2001. (7)

[128] Olivier Markowitch, Dieter Gollmann, and Steve Kremer. On fairness in exchange protocols. In Pil Joong Lee and Chae Hoon Lim, editors, *International Conference on Information Security and Cryptology – ICISC 2002*, volume 2587 of *Lecture Notes in Computer Science*, pages 451–464, Seoul, Korea, November 2002. Springer-Verlag. (11)

[129] Olivier Markowitch and Yves Roggeman. Probabilistic non-repudiation without trusted third party. In *Conference on Security in Communication Networks – SCN'99*, Amalfi, Italy, September 1999. (8)

[130] Olivier Markowitch and Shahrokh Saeednia. Optimistic fair exchange with transparent signature recovery. In Paul Syverson, editor, *Financial Cryptography – FC'01*, volume 2339 of *Lecture Notes in Computer Science*, pages 339–350, Cayman Islands, February 2001. IFCA, Springer-Verlag. (9)

[131] Nele Mentens, Lejla Batina, Bart Preneel, and Ingrid Verbauwhede. Cracking Unix passwords using FPGA platforms. SHARCS - Special Purpose Hardware for Attacking Cryptographic Systems, February 2005. Ecrypt. (133)

[132] Kearns Michael. Efficient noise-tolerant learning from statistical queries. In *ACM Symposium on Theory of Computing – STOC'93*, pages 392–401, San Diego, California, USA, May 1993. ACM, ACM Press. (111)

[133] John Mitsianis. A new approach to enforcing non-repudiation of receipt. Manuscript, 2001. (8)

[134] David Molnar, Andrea Soppera, and David Wagner. A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In Bart Preneel and Stafford Tavares, editors, *Selected Areas in Cryptography – SAC 2005*, Lecture Notes in Computer Science, Kingston, Canada, August 2005. Springer-Verlag. (70, 144)

[135] David Molnar, Andrea Soppera, and David Wagner. A scalable, delegatable, pseudonym protocol enabling ownership transfer of RFID tags. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005. Ecrypt. (144)

[136] David Molnar and David Wagner. Privacy and security in library RFID: Issues, practices, and architectures. In Birgit Pfitzmann and Peng Liu, editors, *Conference on Computer and Communications Security – CCS'04*, pages 210–219, Washington, DC, USA, October 2004. ACM, ACM Press. (70, 77, 78, 105, 109, 113, 115, 117, 122)

[137] Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO'03*, volume 2729 of *Lecture Notes in Computer Science*, pages 617–630, Santa Barbara, California, USA, August 2003. IACR, Springer-Verlag. (125, 126, 131, 132, 134, 136, 138)

[138] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. Cryptographic approach to "privacy-friendly" tags. In *RFID Privacy Workshop*, MIT, Massachusetts, USA, November 2003. (77, 105, 109, 113, 115, 123)

[139] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita. Efficient hash-chain based RFID privacy protection scheme. In *International Conference on Ubiquitous Computing – Ubicomp, Workshop Privacy: Current Status and Future Directions*, Nottingham, England, September 2004. (110)

[140] Tatsuaki Okamoto and Kazuo Ohta. How to simultaneously exchange secrets by general assumptions. In *Conference on Computer and Communications Security – CCS'94*, pages 184–192, Fairfax, Virginia, USA, November 1994. ACM, ACM Press. (8)

[141] Henning Pagnia, Holger Vogt, and Felix Gärtner. Fair exchange. *The Computer Journal*, 46(1):55–75, January 2003. (14)

[142] Torben Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, Santa Barbara, California, USA, August 1991. IACR, Springer-Verlag. (14)

[143] Kenneth Perry and Sam Toueg. Distributed agreement in the presence of processor and communication faults. *IEEE Transactions on Software Engineering*, 12(3):477–482, March 1986. (33, 34)

[144] Philips. I-Code1 Label ICs protocol air interface. `http://www.semiconductors.philips.com`, May 2002. (82)

[145] Jean-Jacques Quisquater and Jean-Paul Delescaille. How easy is collision search? Application to DES (extended summary). In Jean-Jacques Quisquater and Vandewalle Joos, editors, *Advances in Cryptology – EUROCRYPT'89*, volume 434 of *Lecture Notes in Computer Science*, pages 429–434, Houthalen, Belgium, April 1989. IACR, Springer-Verlag. (132)

[146] Michael Rabin. Transaction protection by beacons. *Journal of Computer and System Sciences*, 27(2):256–267, October 1983. (8)

[147] Philippe Raïpin Parvédy and Michel Raynal. Optimal early stopping uniform consensus in synchronous systems with process omission failures. In *ACM Symposium on Parallel Algorithms and Architectures – SPAA'04*, pages 302–310, Barcelona, Spain, June 2004. ACM, ACM Press. (32, 37)

[148] Indrajit Ray and Indrakshi Ray. Fair exchange in e-commerce. *ACM SIGecom Exchanges*, 3(2):9–17, March 2002. (9)

[149] Indrajit Ray, Indrakshi Ray, and Natarajan Narasimhamurthi. A fair-exchange e-commerce protocol with automated dispute resolution. In Bhavani Thuraisingham, Reind van de Riet, Klaus Dittrich, and Zahir Tari, editors, *Annual Working Conference on Database Security – DBSec 2000*, volume 201 of *IFIP Conference Proceedings*, pages 27–38, Schoorl, The Netherlands, August 2000. Kluwer Academic Publishers. (8)

[150] Indrakshi Ray and Indrajit Ray. An optimistic fair exchange e-commerce protocol with automated dispute resolution. In Kurt Bauknecht, Sanjay Kumar Madria, and Günther Pernul, editors, *Electronic Commerce and Web Technologies – EC-Web 2000*, volume 1875 of *Lecture Notes in Computer Science*, pages 84–93, London, United Kingdom, September 2000. DEXA Association, Springer-Verlag. (9)

[151] Keunwoo Rhee, Jin Kwak, Seungjoo Kim, and Dongho Won. Challenge-response based RFID authentication protocol for distributed database environment. In Dieter Hutter and Markus Ullmann, editors, *International Conference on Security in Pervasive Computing – SPC 2005*, volume 3450 of *Lecture Notes in Computer Science*, pages 70–84, Boppard, Germany, April 2005. Springer-Verlag. (105, 107, 108, 109, 113)

[152] SafeTzone. `http://www.safetzone.com`. (63)

[153] Junichiro Saito, Jae-Cheol Ryou, and Kouichi Sakurai. Enhancing privacy of universal re-encryption scheme for RFID tags. In Laurence Jang, Minyi Guo, Guang Gao, and Niraj Jha, editors, *Embedded and Ubiquitous Computing – EUC 2004*, volume 3207 of *Lecture Notes in*

Computer Science, pages 879–890, Aizu-Wakamatsu City, Japan, August 2004. Springer-Verlag. (76, 85, 89, 91, 92)

[154] Tuomas Sandholm and Victor Lesser. Advantages of a leveled commitment contracting protocol. In *National Conference on Artificial Intelligence*, volume 1, pages 126–133, Portland, Oregon, USA, August 1996. AAAI Press. (8)

[155] Sanjay Sarma, Stephen Weis, and Daniel Engels. RFID systems and security and privacy implications. In Burton Kaliski, Çetin Kaya Koç, and Christof Paar, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 454–469, Redwood Shores, California, USA, August 2002. Springer-Verlag. (70)

[156] Sanjay Sarma, Stephen Weis, and Daniel Engels. Radio-frequency identification: security risks and challenges. *Cryptobytes, RSA Laboratories*, 6(1):2–9, Spring 2003. (70)

[157] Naveen Sastry, Umesh Shankar, and David Wagner. Secure verification of location claims. In Douglas Maughan and Adrian Perrig, editors, *ACM Workshop on Wireless Security – WiSe*, pages 1–10, San Diego, California, USA, September 2003. ACM, ACM Press. (71)

[158] Berry Schoenmakers. A simple publicly verifiable secret sharing scheme and its application to electronic voting. In Michael Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 148–164, Santa Barbara, California, USA, August 1999. IACR, Springer-Verlag. (47)

[159] Matthias Schunter. *Optimistic Fair Exchange.* PhD thesis, University of Saarlandes, Saarbruken, Germany, October 2000. (7, 9)

[160] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979. (45, 47)

[161] Dale Skeen. Non-blocking commit protocols. In Edmund Y. Lien, editor, *ACM SIGMOD International Conference on Management of data*, pages 133–142, Ann Arbor, Michigan, USA, April–May 1981. ACM, ACM Press. (32, 34)

[162] Markus Stadler. Publicly verifiable secret sharing. In Ueli Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 190–199, Saragossa, Spain, May 1996. IACR, Springer-Verlag. (45, 47, 48, 49)

[163] François-Xavier Standaert, Gael Rouvroy, Jean-Jacques Quisquater, and Jean-Didier Legat. A time-memory tradeoff using distinguished points: New analysis & FPGA results. In Burton Kaliski, Çetin Kaya Koç, and Christof Paar, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2002*, volume 2523 of *Lecture Notes in Computer Science*, pages 593–609, Redwood Shores, California, USA, August 2002. Springer-Verlag. (132, 135)

[164] Paul Syverson. Weakly secret bit commitment: Applications to lotteries and fair exchange. In *IEEE Computer Security Foundations Workshop – CSFW'98*, pages 2–13, Rockport, Massachusetts, USA, June 1998. IEEE. (8)

[165] Tom Tedrick. How to exchange half a bit. In David Chaum, editor, *Advances in Cryptology – CRYPTO'83*, pages 147–151. Plenum Press, August 1983. (8)

[166] Tom Tedrick. Fair exchange of secrets. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 434–438, Santa Barbara, California, USA, August 1985. IACR, Springer-Verlag. (8)

[167] J. Toonstra and Wintold Kinsner. Transient analysis and genetic algorithms for classification. In *IEEE WESCANEX 95. Communications, Power, and Computing*, volume 2, pages 432–437, Winnipeg, Manitoba, Canada, May 1995. IEEE Computer Society Press. (84)

[168] Trusted Computing Group. Trusted computing group homepage. `https://www.trustedcomputinggroup.org`, 2003. (27)

[169] Mitsuo Usami. An ultra small RFID chip:μ-chip. In *Asia-Pacific Conference on Advanced System Integrated Circuits – AP-ASIC 2004*, pages 2–5, Fukuoka, Japan, August 2004. IEEE. (63)

[170] George Varghese and Nancy Lynch. A tradeoff between safety and liveness for randomized coordinated attack. In *ACM symposium on Principles of Distributed Computing – PODC'92*, pages 241–250, Vancouver, British Columbia, Canada, August 1992. ACM, ACM Press. (22)

[171] George Varghese and Nancy Lynch. A tradeoff between safety and liveness for randomized coordinated attack. *Information and Computation*, 128(1):57–71, July 1996. (22)

[172] Holger Vogt. Asynchronous optimistic fair exchange based on revocable items. In Rebecca Wright, editor, *Financial Cryptography – FC'03*, volume 2742 of *Lecture Notes in Computer Science*, Le Gosier, Guadeloupe, French West Indies, January 2003. IFCA, Springer-Verlag. (9, 14)

[173] Holger Vogt, Felix Gärtner, and Henning Pagnia. Supporting fair exchange in mobile environments. *Journal on Mobile Networks and Applications*, 8(2):127–136, April 2003. (14, 19)

[174] Holger Vogt, Henning Pagnia, and Felix Gärtner. Using smart cards for fair exchange. In Ludger Fiege, Gero Mühl, and Uwe Wilhelm, editors, *International Workshop on Electronic Commerce – WELCOM'01*, volume 2232 of *Lecture Notes in Computer Science*, pages 101–113, Heidelberg, Germany, November 2001. Springer-Verlag. (14, 19)

[175] Steve Weingart. Physical security devices for computer subsystems: A survey of attacks and defenses. In Çetin Kaya Koç and Christof Paar, editors, *Workshop on Cryptographic Hardware and Embedded Systems – CHES 2000*, volume 1965 of *Lecture Notes in Computer Science*, pages 302–317, Worcester, Massachusetts, USA, August 2000. Springer-Verlag. (63)

[176] Stephen Weis. Security and privacy in radio-frequency identification devices. Master thesis, Massachusetts Institute of Technology (MIT), Massachusetts, USA, May 2003. (70, 78)

[177] Stephen Weis. Security parallels between people and pervasive devices. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pages 105–109, Kauai Island, Hawaii, USA, March 2005. IEEE, IEEE Computer Society Press. (111)

[178] Stephen Weis, Sanjay Sarma, Ronald Rivest, and Daniel Engels. Security and privacy aspects of low-cost radio frequency identification systems. In Dieter Hutter, Günter Müller, Werner Stephan, and Markus Ullmann, editors, *International Conference on Security in Pervasive Computing – SPC 2003*, volume 2802 of *Lecture Notes in Computer Science*, pages 454–469, Boppard, Germany, March 2003. Springer-Verlag. (70, 105, 107, 109, 113)

[179] Doug Whiting, Russ Housley, and Niels Ferguson. Counter with CBC-MAC (CCM), September 2003. IETF – RFC 3610. (20)

[180] Michael Wiener and Paul van Oorschot. Parallel collision search with cryptanalytic applications. *Journal of Cryptology*, 12(1):1–28, March 1999. (132)

[181] Jeongkyu Yang, Jaemin Park, Hyunrok Lee, Kui Ren, and Kwangjo Kim. Mutual authentication protocol for low-cost RFID. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005. Ecrypt. (63, 85, 103)

[182] Tatu Ylonen. SSH transport layer protocol, March 2005. IETF Internet Draft. (20)

[183] Ning Zhang and Qi Shi. Achieving non-repudiation of receipt. *The Computer Journal*, 39(10):844–853, 1996. (8)

[184] Ning Zhang, Qi Shi, and Madjid Merabti. A flexible approach to secure and fair document exchange. *The Computer Journal*, 42(7):569–581, 1999. (16)

[185] Xiaolan Zhang and Brian King. Integrity improvements to an RFID privacy protection protocol for anti-counterfeiting. In Jianying Zhou, Javier Lopez, Robert Deng, and Feng Bao, editors, *Information Security Conference – ISC 2005*, volume 3650 of *Lecture Notes in Computer Science*, pages 474–481, Singapore, September 2005. Springer-Verlag. (103)

[186] Jianying Zhou. *Non-Repudiation*. PhD thesis, University of London, London, Great Britain, December 1996. (7, 8, 14)

[187] Jianying Zhou, Robert Deng, and Feng Bao. Evolution of fair non repudiation with TTP. In Josef Pieprzyk, Reihaneh Safavi-Naini, and Jennifer Seberry, editors, *Australasian Conference on Information Security and Privacy – ACISP'99*, volume 1587 of *Lecture Notes in Computer Science*, pages 258–269, Wollongong, Australia, April 1999. Springer-Verlag. (9)

[188] Jianying Zhou, Robert Deng, and Feng Bao. Some remarks on a fair exchange protocol. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography – PKC'00*, volume 1751 of *Lecture Notes in Computer Science*, pages 46–57, Melbourne, Australia, January 2000. Springer-Verlag. (9)

[189] Jianying Zhou and Dieter Gollmann. Certified electronic mail. In Elisa Bertino, Helmut Kurth, Giancarlo Martella, and Emilio Montolivo, editors, *European Symposium on Research in Computer Security – ESORICS 1996*, volume 1146 of *Lecture Notes in Computer Science*, pages 160–171, Rome, Italia, September 1996. Springer-Verlag. (8)

[190] Jianying Zhou and Dieter Gollmann. A fair non-repudiation protocol. In *IEEE Symposium on Research in Security and Privacy*, pages 55–61, Oakland, California, USA, May 1996. IEEE, IEEE Computer Society Press. (8)

Curriculum Vitæ

Education

- **PhD**,
  Supervisor: Prof. Serge Vaudenay,
  Swiss Federal Institute of Technology, Switzerland.
- **Graduate School in Communication Systems**,
  Supervisor: Prof. Serge Vaudenay,
  Swiss Federal Institute of Technology, Switzerland.
- **Master of Computer Science**,
  Supervisor: Prof. Brigitte Vallée,
  French designation: Maîtrise et DEA d'informatique,
  University of Caen, France.
- **Bachelor of Computer Science**,
  French designation: Licence d'informatique,
  University of Caen, France.
- **Bachelor of Mathematics**,
  French designation: Licence de mathématiques,
  University of Caen, France.

Awards

- Best student paper award obtained for *Privacy Issues in RFID Banknote Protection Schemes* at CARDIS 2004.
- Top one ranking of the *maîtrise d'informatique* (fourth year diploma), University of Caen, 1999.

GILDAS AVOINE

Books

- Gildas Avoine, Pascal Junod, and Pascal Oechslin. *Sécurité informatique – Exercices corrigés (Collection of exercises in computer security).* Vuibert, Paris, France, 2004.
- Serge Vaudenay, Gildas Avoine, and Pascal Junod. *Cryptographie, théorie et pratique, French translation of the second edition of "Cryptography, theory and practice" (D. Stinson).* Vuibert, Paris, France, second edition, 2003.

Papers with Peer Review

- Gildas Avoine, Pascal Junod, and Philippe Oechslin. Time-memory trade-offs: False alarm detection using checkpoints. In *Progress in Cryptology – Indocrypt 2005*, volume 3797 of *Lecture Notes in Computer Science*, pages 183–196, Bangalore, India, December 2005. Cryptology Research Society of India, Springer-Verlag.
- Gildas Avoine, Etienne Dysli, and Philippe Oechslin. Reducing time complexity in RFID systems. In *Selected Areas in Cryptography – SAC 2005*, Lecture Notes in Computer Science, Kingston, Canada, August 2005. Springer-Verlag.
- Gildas Avoine, Felix Gärtner, Rachid Guerraoui, and Marko Vukolić. Gracefully degrading fair exchange with security modules. In *European Dependable Computing Conference – EDCC-5*, volume 3463 of *Lecture Notes in Computer Science*, pages 55–71, Budapest, Hungary, April 2005. Springer-Verlag.
- Gildas Avoine and Philippe Oechslin. A scalable and provably secure hash based RFID protocol. In *International Workshop on Pervasive Computing and Communication Security – PerSec 2005*, pages 110–114, Kauai Island, Hawaii, USA, March 2005. IEEE, IEEE Computer Society Press.
- Gildas Avoine and Philippe Oechslin. RFID traceability: A multilayer problem. In *Financial Cryptography – FC'05*, volume 3570 of *Lecture Notes in Computer Science*, pages 125–140, Roseau, The Commonwealth Of Dominica, February–March 2005. IFCA, Springer-Verlag.
- Gildas Avoine. Fraud within asymmetric multi-hop cellular networks. In *Financial Cryptography – FC'05*, volume 3570 of *Lecture Notes in Computer Science*, pages 1–15, Roseau, The Commonwealth Of Dominica, February–March 2005. IFCA, Springer-Verlag.
- Gildas Avoine, Jean Monnerat, and Thomas Peyrin. Advances in alternative non-adjacent form representations. In *Progress in Cryptology – Indocrypt 2004*, volume 3348 of *Lecture Notes in Computer Science*, pages 260–274, Chennai, India, December 2004. Cryptology Research Society of India, Springer-Verlag.
- Gildas Avoine. Privacy issues in RFID banknote protection schemes. In *International Conference on Smart Card Research and Advanced Applications – CARDIS*, pages 33–48, Toulouse, France, August 2004. IFIP, Kluwer Academic Publishers.
- Gildas Avoine and Serge Vaudenay. Optimistic fair exchange based on publicly verifiable secret sharing. In *Australasian Conference on Information Security and Privacy*

– *ACISP'04*, volume 3108 of *Lecture Notes in Computer Science*, pages 74–85, Sydney, Australia, July 2004. Springer-Verlag.

- Gildas Avoine and Serge Vaudenay. Fair exchange with guardian angels. In *International Workshop on Information Security Applications – WISA 2003*, volume 2908 of *Lecture Notes in Computer Science*, pages 188–202, Jeju Island, Korea, August 2003. Springer-Verlag.
- Gildas Avoine and Serge Vaudenay. Cryptography with guardian angels: Bringing civilization to pirates. *Report on a Working Session on Security in Wireless Ad Hoc Networks, Levente Buttyán and Jean-Pierre Hubaux editors, ACM Mobile Computing and Communications Review*, 7(1):74–94, January 2003.

Technical Reports

- Gildas Avoine, Pascal Junod, and Philippe Oechslin. Time-memory trade-offs: False alarm detection using checkpoints. Technical Report LASEC-REPORT-2005-002, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, September 2005.
- Gildas Avoine. Adversary model for radio frequency identification. Technical Report LASEC-REPORT-2005-001, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, September 2005.
- Gildas Avoine, Felix Gärtner, Rachid Guerraoui, and Marko Vukolić. Gracefully degrading fair exchange with security modules. Technical Report EPFL-I&C 26, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, March 2004.
- Gildas Avoine, Felix Gärtner, Rachid Guerraoui, Klaus Kursawe, Serge Vaudenay, and Marko Vukolić. Reducing fair exchange to atomic commit. Technical Report EPFL-I&C 11, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, February 2004.

Articles in Magazines

- Gildas Avoine. Introduction aux protocoles d'échange équitable (Introduction to fair exchange). *Multi-System & Internet Security Cookbook – MISC*, (4):10–11, November 2002.
- Gildas Avoine and Pascal Junod. PGP : comment éviter les mauvaises surprises ? (PGP: How to avoid nasty surprises?). *Multi-System & Internet Security Cookbook – MISC*, (3):92–97, July 2002.

Presentations

- Time-memory trade-offs: False alarm detection using checkpoints. Conference Indocrypt, December 2005, Bangalore, India.
- La technologie RFID: un don de Dieu ou un défi du Diable ? INRIA (seminar), November 2005, Grenoble, France.

- Open questions in radio frequency identification. MyCrypt, September 2005, Kuala Lumpur, Malaysia.
- Contact-less specificity in term of security and RFID security. Smart University (Tutorial), September 2005, Sophia-Antipolis, France.
- Scalability issues in RFID systems. Ecrypt Workshop on RFID and Lightweight Crypto (Invited talk), July 2005, Graz, Austria.
- Time-memory trade-offs: False alarm detection using checkpoints. EUROCRYPT'05 (Rump Session), May 2005, Århus, Denmark.
- RFID security issues. EPFL, Course on Selected Topics in Cryptography (lecture), April 2005, Lausanne, Switzerland.
- A scalable and provably secure hash-based RFID protocol. Workshop PerSec 2005, March 2005, Kauai Island, Hawaii, USA.
- Fraud within asymmetric multi-hop cellular networks. Conference Financial Cryptography – FC'05, March 2005, Roseau, Dominica.
- RFID traceability: A multilayer problem. Conference Financial Cryptography – FC'05, March 2005, Roseau, Dominica.
- Traçabilité dans les systèmes RFID. Journées Codage et Cryptographie, February 2005, Aussois, France.
- RFID: A new challenge for cryptographers? University of Caen, LMNO/GREYC (Seminar), January 2005, Caen, France.
- Advances in alternative non-adjacent form representations. Conference Indocrypt, December 2004, Chennai, India.
- Security issues in RFID banknote protection schemes. Conference Cardis, August 2004, Toulouse, France.
- How to design an RFID protocol? CRYPTO'04 (Rump Session), August 2004, Santa Barbara, California, USA.
- Optimistic fair exchange based on publicly verifiable secret sharing. Conference ACISP, July 2004, Sydney, Australia.
- Radio frequency identification systems in our daily lives. Workshop on Mobile Information and Communication Systems, July 2004, Zurich, Switzerland.
- Security issues in RFID systems. EPFL, course on Security Protocols and Applications (lecture), May 2004, Lausanne, Switzerland.
- Fair exchange with guardian angels. Workshop on Mobile Information and Communication Systems, October 2003, Ascona, Switzerland.
- Fair exchange with guardian angels. Workshop on Information Security Applications – WISA'03, August 2003, Jeju Island, Korea.
- Fair exchange with observers. French-Swiss cryptographers meeting, April 2003, Crans-Montana, Switzerland.