

# CRYPTOGRAPHIC PROTOCOL DESIGN

Sven Laur

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Information and Natural Sciences for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 25th of April, 2008, at 12 noon.

Helsinki University of Technology  
Faculty of Information and Natural Sciences  
Department of Information and Computer Science

Teknillinen korkeakoulu  
Informaatio- ja luonnontieteiden tiedekunta  
Tietojenkäsittelytieteen laitos

Distribution:

Helsinki University of Technology  
Faculty of Information and Natural Sciences  
Department of Information and Computer Science  
P.O.Box 5400  
FI-02015 TKK  
FINLAND  
URL: <http://ics.tkk.fi>  
Tel. +358 9 451 3264  
Fax. +358 9 451 3369  
E-mail: [series@ics.tkk.fi](mailto:series@ics.tkk.fi)

© Sven Laur

ISBN 978-951-22-9292-9 (Print)  
ISBN 978-951-22-9293-6 (Online)  
ISSN 1797-5050 (Print)  
ISSN 1797-5069 (Online)  
URL:<http://lib.tkk.fi/Diss/2008/isbn9789512292936/>

Multiprint Oy  
Espoo 2008

**ABSTRACT:** In this work, we investigate the security of interactive computations. The main emphasis is on the mathematical methodology that is needed to formalise and analyse various security properties. Differently from many classical treatments of secure multi-party computations, we always quantify security in exact terms. Although working with concrete time bounds and success probabilities is technically more demanding, it also has several advantages. As all security guarantees are quantitative, we can always compare different protocol designs. Moreover, these security guarantees also have a clear economical interpretation and it is possible to compare cryptographic and non-cryptographic solutions. The latter is extremely important in practice, since cryptographic techniques are just one possibility to achieve practical security. Also, working with exact bounds makes reasoning errors more apparent, as security proofs are less abstract and it is easier to locate false claims.

The choice of topics covered in this thesis was guided by two principles. Firstly, we wanted to give a coherent overview of the secure multi-party computation that is based on exact quantification of security guarantees. Secondly, we focused on topics that emerged from the author's own research. In that sense, the thesis generalises many methodological discoveries made by the author.

As surprising as it may seem, security definitions and proofs mostly utilise principles of hypothesis testing and analysis of stochastic algorithms. Thus, we start our treatment with hypothesis testing and its generalisations. In particular, we show how to quantify various security properties, using security games as tools. Next, we review basic proof techniques and explain how to structure complex proofs so they become easily verifiable. In a nutshell, we describe how to represent a proof as a game tree, where each edge corresponds to an elementary proof step. As a result, one can first verify the overall structure of a proof by looking at the syntactic changes in the game tree and only then verify all individual proof steps corresponding to the edges.

The remaining part of the thesis is dedicated to various aspects of protocol design. Firstly, we discuss how to formalise various security goals, such as input-privacy, output-consistency and complete security, and how to choose a security goal that is appropriate for a specific setting. Secondly, we also explore alternatives to exact security. More precisely, we analyse connections between exact and asymptotic security models and rigorously formalise a notion of subjective security. Thirdly, we study in which conditions protocols preserve their security guarantees and how to safely combine several protocols. Although composability results are common knowledge, we look at them from a slightly different angle. Namely, it is irrational to design universally composable protocols at any cost; instead, we should design computationally efficient protocols with minimal usage restrictions. Thus, we propose a three-stage design procedure that leads to modular security proofs and minimises usage restrictions.

**KEYWORDS:** asymptotic security, data authentication, exact security, homomorphic encryption, secure multi-party computation, sequential composability, subjective security, time-stamping, universal composability.



**TIIVISTELMÄ:** Tässä työssä tutkitaan vuorovaikutteisen laskennan turvallisuutta. Erityisesti painotetaan matemaattisia menetelmiä, joita tarvitaan erilaisten turvallisuusominaisuuksien määrittelyyn ja analysointiin. Perinteisistä käsittelyta-voista poiketen usean osapuolen laskennan turvallisuutta mitataan tässä työssä tarkoilla suureilla. Vaikka tarkkojen rajojen käyttö arvioitaessa laskennallista vaativuutta ja onnistumisen todennäköisyyttä on teknisesti vaativampaa, sillä on myös useita etuja. Se tekee mahdolliseksi eri protokollien välisen vertailun. Lisäksi, tällaisilla turvallisuuden mitoilla on selkeä kustannustaloudellinen tul- kinta, mikä tekee mahdolliseksi vertailla salaustekniikkaa käyttävien ja muiden tietoturvallisuusratkaisujen kustannuksia. Tällä on merkitystä käytännön kannal- ta, koska salaustekniikkaan perustuvat menetelmät ovat usein vain yksi vaihto- ehto käytännön turvallisuusjärjestelmiä toteutettaessa. Lisäksi tarkkojen rajo- jen käyttö tekee turvallisuustodistuksista selkeämpiä ja siten helpottaa todistusten päättelyvirheiden havaitsemista.

Tämän työn aiheiden valinta perustuu kahteen periaatteeseen. Ensimmäisen mukaan tavoitteena on luoda johdonmukainen katsaus usean osapuolen lasken- nan turvallisuuteen, joka perustuu turvallisuustakuiden tarkkaan määrittämiseen. Toisen periaatteen mukaan keskitytään tarkastelemaan aiheita, jotka ovat olleet tekijän tutkimusten kohteena. Tässä väitöskirjassa esitetään yleistetyssä muodossa monia tekijän tekemiä menetelmiä koskevia löydöksiä.

Niin yllättävältä kuin se tuntuukin, turvallisuuden määritelmässä ja todistuk- sissa käytetään tilastollisen päättelyn ja stokastisten algoritmien menetelmiä. Siksi tämän työn aluksi tarkastellaan hypoteesien testausta ja sen yleistyksiä. Erityi- sesti osoitetaan kuinka erilaisille turvallisuusominaisuuksille voidaan antaa nu- meerinen arvo turvallisuuspelejä käyttäen. Seuraavaksi tarkastellaan todistuksen perustekniikoita ja esitetään kuinka todistus tulee rakentaa, jotta se on helposti todennettavissa. Kiteytettynä tämä tarkoittaa, että kuvataan turvallisuuspelejä puu- na, jonka jokainen kaari vastaa yksinkertaista askelta todistuksessa. Näin esitetty todistus voidaan todentaa tarkastelemalla ensin sen syntaktista kokonaisraken- netta ja sen jälkeen todentamalla jokaista puun kaarta vastaava todistusaskel.

Väitöskirjan loppuosassa tarkastellaan salausteknisten protokollien suunnit- telun eri piirteitä. Ensiksi käsitellään erilaisten turvallisuustavoitteiden, kuten syötteen yksityisyys, tuotoksen oikeellisuus ja täydellinen turvallisuus, täsmällistä määrittelyä ja sitä, kuinka turvallisuustavoite tulee asettaa vastaamaan konkreet- tista tilannetta. Toiseksi tutkitaan tarkan turvallisuuden vaihtoehtoja. Tarkem- min sanottuna analysoidaan tarkan ja asymptoottisen turvallisuusmallin välisiä yhteyksiä ja annetaan täsmällinen määritelmä subjektiiviselle turvallisuudelle. Kolmanneksi tarkastellaan ehtoja, joilla protokolla säilyttää turvallisuusomina- suutensa, ja kuinka useita protokollia voidaan yhdistää turvallisesti. Salausteknis- ten protokollien teoriassa yhdistettävyyttä koskevat tulokset tunnetaan yleisesti, mutta tässä työssä niitä tarkastellaan uudesta näkökulmasta. Nimittäin, ei ole mielekäästä rakentaa universaalisti yhdisteltävissä olevia protokollia mihin hin- taan hyvänsä, vaan tuloksena olevien protokollien tulee olla tehokkaita ja käytön rajoitusten niin pieniä kuin mahdollista. Tässä työssä esitetään kolmivaiheinen menettely, jolla saavutetaan modulaariset turvallisuustodistukset ja minimaaliset käytön rajoitukset.

**AVAINSANAT:** asymptoottinen ja tarkka turvallisuus, datan autentikointi, ho- momorfinen salaus, turvallinen usean osapuolen välinen laskenta, peräkkäinen yhdistettävyyden, subjektiivinen turvallisuus, aikaleimaus, universaali yhdistettävyyden.



# CONTENTS

|          |                                                              |           |
|----------|--------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                          | <b>1</b>  |
| 1.1      | Cryptography as an Engineering Discipline . . . . .          | 2         |
| 1.2      | Introduction to Cryptographic Protocol Design . . . . .      | 4         |
| 1.3      | Benefits of Finite Set Policy . . . . .                      | 7         |
| 1.4      | Contributions of the Author . . . . .                        | 8         |
| <b>2</b> | <b>Common Notation and Basic Concepts</b>                    | <b>12</b> |
| 2.1      | Basic Mathematical Concepts . . . . .                        | 12        |
| 2.2      | Different Interpretations of Probability . . . . .           | 14        |
| 2.3      | Basic Properties of Random Variables . . . . .               | 15        |
| 2.4      | Different Formal Models of Computation . . . . .             | 17        |
| <b>3</b> | <b>Hypothesis Testing</b>                                    | <b>22</b> |
| 3.1      | Simple Hypothesis Testing . . . . .                          | 22        |
| 3.2      | Negligible Events and Semantic Security . . . . .            | 24        |
| 3.3      | Interactive Inference and Security Games . . . . .           | 26        |
| <b>4</b> | <b>Cryptographic Proof Techniques</b>                        | <b>31</b> |
| 4.1      | Reductions As Rewriting Rules . . . . .                      | 32        |
| 4.2      | Reductions and Time-success Profiles . . . . .               | 33        |
| 4.3      | Surprising Properties of Conditional Probabilities . . . . . | 36        |
| 4.4      | From Game Chains to Proof Trees . . . . .                    | 38        |
| 4.5      | Formal Verification of Cryptographic Proofs . . . . .        | 41        |
| <b>5</b> | <b>Security of Interactive Computations</b>                  | <b>45</b> |
| 5.1      | Formal Requirements to Security Definitions . . . . .        | 46        |
| 5.2      | Security of Idealised Computations . . . . .                 | 47        |
| 5.3      | The Real versus Ideal World Paradigm . . . . .               | 51        |
| 5.4      | Security in Semi-Honest Model . . . . .                      | 55        |
| 5.5      | Input-Privacy in Malicious Model . . . . .                   | 56        |
| 5.6      | Output-Consistency in Malicious Model . . . . .              | 57        |
| 5.7      | Complete Security in Malicious Model . . . . .               | 59        |
| 5.8      | Canonical Constructive Correspondence . . . . .              | 60        |
| <b>6</b> | <b>Alternative Security Models</b>                           | <b>64</b> |
| 6.1      | Scalability and Polynomial Security . . . . .                | 64        |
| 6.2      | Asymptotic Security for Protocols . . . . .                  | 68        |
| 6.3      | Gentle Introduction to Subjective Security . . . . .         | 72        |
| 6.4      | Setup Assumptions and Subjective Security . . . . .          | 74        |
| 6.5      | Rigorous Formalisation of Subjective Security . . . . .      | 77        |
| 6.6      | Reductions and Subjective Security Premises . . . . .        | 80        |
| 6.7      | Strictly Constructive Proof Techniques . . . . .             | 83        |

|          |                                                         |            |
|----------|---------------------------------------------------------|------------|
| <b>7</b> | <b>Modular Design of Complex Protocols</b>              | <b>86</b>  |
| 7.1      | Duality between Protocol Design and Analysis . . . . .  | 87         |
| 7.2      | Layered Description of Computational Contexts . . . . . | 89         |
| 7.3      | Two Flavours of Stand-Alone Security . . . . .          | 92         |
| 7.4      | Canonical Decomposition Techniques . . . . .            | 95         |
| 7.5      | Characterisation of Sequential Composability . . . . .  | 97         |
| 7.6      | Characterisation of Universal Composability . . . . .   | 102        |
| 7.7      | Trusted Setup and Universal Composability . . . . .     | 107        |
| 7.8      | Public Inputs and Universal Composability . . . . .     | 111        |
|          | <b>Bibliography</b>                                     | <b>114</b> |
|          | <b>Index</b>                                            | <b>129</b> |

## PREFACE

This thesis is a result of studies and research at the Laboratory for Theoretical Computer Science of Helsinki University of Technology. I remember vividly my first days in Finland when I was a master student working under the supervision of Helger Lipmaa. Much has changed during my stay in Finland, but the TCS lab has always been a great place to work.

Although these last four years have been quite intense and sometimes even overly stressful, I have learnt a lot during my studies. For that I am indebted to my supervisors Helger Lipmaa and Kaisa Nyberg. They gave out enough puzzles and problems to keep me intrigued. At the same time, they were always helpful and taught me many secrets of research, scientific writing and publishing. They also helped me to get funding from the Academy of Finland and managed the corresponding CRYDAMI research project.

Besides my supervisors, I had pleasure to work together with many other beautiful minds. I am especially grateful to my co-authors Ahto Buldas, Bart Goethals, Emilia Käsper, Taneli Mielikäinen and Sylvain Pasini, who produced many excellent ideas and were patient enough to work with me. I would also like to thank Pekka Orponen and Jaakko Hollmén for their excellent book suggestions, which have made my literature exploration phases much easier.

Also, it would have been much more difficult to write the thesis without comments from demanding readers. For that I am grateful to Billy Brumley, Ahto Buldas, Emilia Käsper, Phillip Rogaway, Berry Schoenmakers and Jan Willemson, who took the time to read the drafts of this thesis, raised many interesting questions, and pointed out many errors and inconsistencies.

Last but not least, I would like to thank my family and friends from Tartu, Otaniemi and Espoo, who kept me in balance during tough times.

Tartu, March 2008

Sven Laur

## CONTRIBUTIONS OF THE AUTHOR

This thesis consists of a mini-monograph followed by four articles. The articles were carefully selected out of seven articles published by the author to illustrate the theoretical methodology developed in the first part of the thesis. In short, this thesis gives a systematic overview of secure multi-party computations in the framework of exact security. To the author's knowledge, no similar work has been published in cryptographic literature, although the need for such treatment is apparent. We emphasise that the first part of the thesis is not just a summary but contains several technical novelties and methodological advances:

- We develop a consistent methodology for quantifying the security of interactive computations. The corresponding security definitions are much more descriptive than classical asymptotic security definitions.
- We show that not all security proofs can be formalised as game chains and explain how game trees and new deduction rules solve this problem. Additionally, we describe how to verify the overall structure of a proof by looking at the syntactic changes in the game tree.
- We are the first to consistently use time-success and risk profiles to characterise quantitative properties of cryptographic reductions.
- We explicitly discuss intermediate levels of security between security in the semi-honest model and complete security in the malicious model. Such models provide cost-efficient solutions for many practical tasks.
- We establish a rigorous formalisation of subjective security. As a result, one can formulate and prove security claims for individual public keys and hash functions and thus escape limitations of classical formalisations.
- Finally, we show how to combine protocols with different composability guarantees. The corresponding three-stage design methodology can significantly simplify the design of complex protocols.

The included articles [P1, P2, P3, P4] provide solutions to specific but practically important computational tasks and thus nicely illustrate different theoretical aspects covered in the first part of the thesis.

[P1]: Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On Private Scalar Product Computation for Privacy-Preserving Data Mining. In Choonsik Park and Seongtaek Chee, editors, *Information Security and Cryptology - ICISC 2004, 7th International Conference, Seoul, Korea, December 2-3, 2004, Revised Selected Papers*, volume 3506 of *Lecture Notes in Computer Science*, pages 104–120. Springer, 2004.

[P2]: Sven Laur and Helger Lipmaa. A New Protocol for Conditional Disclosure of Secrets and Its Applications. In Jonathan Katz and Moti Yung, editors, *Applied Cryptography and Network Security, 5th International Conference, ACNS 2007, Zhuhai, 5-8 June, 2007*, volume 4521 of *Lecture Notes in Computer Science*, pages 207–225. Springer, 2007.

- [P3]: Ahto Buldas and Sven Laur. Knowledge-Binding Commitments with Applications in Time-Stamping. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *The International Conference on Theory and Practice of Public-Key Cryptography, PKC 2007, Beijing, 16-20 April, 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 150–165. Springer, 2007.
- [P4]: Sven Laur and Kaisa Nyberg. Efficient Mutual Data Authentication Using Manually Authenticated Strings. In David Pointceval, Yi Mu, and Kefei Chen, editors, *The 5th International Conference on Cryptology and Network Security, CANS 2006, Suzhou, Dec. 8-10, 2006*, volume 4301 of *Lecture Notes in Computer Science*, pages 90–107. Springer, 2006.

The author has made significant contributions to all of the abovementioned articles. In short, the author derived most proofs and attacks in the article [P1]. The basic methodology used in the article [P2] was also discovered by the author, although the current exposition is a result of extensive joint work with H. Lipmaa. Similarly, the author established the notion of knowledge-binding commitments and corresponding reductions in the article [P3]. Finally, the original problem statement and the initial solution in the article [P4] is due to K. Nyberg and N. Asokan. However, the author derived all security proofs and made several technical refinements that were essential for achieving provable security.



# 1 INTRODUCTION

Good design is always a trade-off between conflicting requirements regardless of whether we talk about physical objects, such as buildings and devices, or mental constructions, such as mathematical objects and computer programs. Some of those design constraints are easy to describe and formalise, whereas others are either subjective or inexplicable. Therefore, the design process itself is often a mixture of creative and engineering steps and thus not completely expressible using formal methodology. Nevertheless, we can use mathematical models as powerful tools to formalise design requirements, to limit the search space and to validate the resulting prototypes against objective constraints.

In this thesis, we describe how to formalise and analyse the security of distributed computations. Since a computational process is always inseparable from physical phenomena that make it observable, simplified mathematical models, such as finite automata and Turing machines have their limitations. Namely, some artefacts that are observable in theoretical models might not be present or meaningful in practice. Therefore, we must make a constant conscious effort to assure that theoretical models adequately represent the reality.

In particular, note that security as a concept cannot be viewed as a thing in itself, rather it has a meaning only in a larger economical, political or military context, where entities and organisations compete against each other. Consequently, these entities have to withstand internal and external attacks in order to achieve their goals. As reliable information transfer and processing are usually among the key factors to success, we have to protect computations against wire-tapping and device corruption. Essentially, there are three main types of protection methods: physical, organisational and cryptographic countermeasures. Physical countermeasures limit the abilities of a potential attacker to compromise critical computational processes. For example, servers that run mission critical software are often located in physically guarded rooms. Organisational countermeasures change the common process logic in order to demotivate potential attackers, limit potential damage, or provide fail-safe recovery procedures. For example, making regular back-ups from mission critical data can both limit the potential damage and simplify the recovery procedure.

Cryptographic countermeasures are somewhat different, since they modify the computational process itself and not the surrounding context. Although this approach can be immensely flexible, there are certain inherent limitations. Most importantly, cryptographic methods are applicable only in distributed settings. Indeed, if an adversary gets full control over the computational process, then he or she can ignore all cryptographic countermeasures built into the process. Moreover, successful recovery from such attacks is impossible, as the adversary can corrupt the entire internal state. In particular, note that back-up strategies make sense only in a distributed setting, otherwise the adversary can corrupt the back-ups, too. To minimise such risks, mission-critical processes are often divided into well-isolated sub-processes or accompanied with unerasable logging procedures. But the latter does not automatically imply that we need many computational devices. In fact, we can still use a single device for all computations, provided that a potential adversary cannot corrupt the entire device and all sub-tasks look like different computational entities.

We emphasise that one cannot achieve security without combining various countermeasures against the attacks. In fact, one must find a proper balance between various protection methods, otherwise a single weakness can make the entire defence meaningless. In short, cryptographic countermeasures achieve their purpose when attacks against non-cryptographic measures are more attractive to the adversary. In many cases, the cost of cryptographic countermeasures is marginal compared to the other measures and thus people tend to use them as cheap substitutes for real solutions. Although cryptographic constructions must have an appropriate safety margin, it is impractical and wasteful to make them overly strong compared to the other protection measures that users are willing to accept. For the same reason, the total cost of countermeasures must match the potential losses and sometimes the best strategy is to ignore threats.

To summarise, finding an appropriate balance between different protection methods is mostly an economical decision that should be based on approximate cost estimates. Unfortunately, precise estimation of potential losses and gains currently resembles more to magic than science. In fact, we still cannot answer even the simplest questions like what would it cost to break a modern 128-bit block cipher, or when the use of 1024-bit RSA becomes a considerable liability issue in e-commerce. Although such a sad state of affairs is really discouraging, we should still try to capture practical requirements in theoretical models, or at least try to develop solutions that are meaningful in practice.

## 1.1 CRYPTOGRAPHY AS AN ENGINEERING DISCIPLINE

Cryptography as a general term unites a wide spectrum of different subfields ranging from empirical design of various base constructions to branches of pure mathematics that focus on complexity theoretical aspects and fundamental limitations in cryptography. Hence, there are many valid viewpoints and possible research goals. In this thesis, we view cryptography mostly as an engineering discipline that should provide formal methodology for making economically sound and scientifically justified decisions. More precisely, we study how to design protocols with desired properties using cryptographic primitives and how the achievable security level depends on the properties of basic primitives. We are mostly interested in the *exact* quantification of security and thus state all security guarantees in terms of concrete running times and success probabilities. Although such a quantification does not directly reflect the corresponding monetary losses and gains, we can easily overcome this by estimating the cost of computations and the perceived utility of a successful attack. The corresponding cost estimation phase is beyond our scope, since it is a purely economical problem. Nevertheless, the existence of such transformations is important, as it gives a clear interpretation to all security estimates and allows us to compare cryptographic and non-cryptographic countermeasures.

In many cases, it is beneficial to combine cryptographic and non-cryptographic countermeasures. Consequently, our formalism must be flexible enough to model a wide range of security goals and adversarial powers. Moreover, none of these theoretical models is a priori better than the others, since the corresponding costs and gains depend on practical details. A solution that is tuned for a particular setting may be completely unsuitable for other settings. As a result,

there are literally hundreds, not to say thousands, of meaningful security models. Since we cannot treat each model separately, we must limit ourselves to the most important aspects that are shared by most of the models.

In particular, it makes sense to use the same formal model of interactive computations. Although all standard formalisations are based on Turing machines, there are some technical details that make Random Access machines more appealing. We remark here that the difference is only in exact resource consumption and both models are qualitatively equivalent, see Chapter 2. Also, note that security properties are commonly described in terms of computational indistinguishability or advantage. Both notions have their origins in statistics. Chapter 3 covers essential properties of hypothesis testing that are necessary to quantify and analyse various security properties. Moreover, the knowledge of these basic concepts is sufficient to carry out most common security proofs as long as we abstract away all complexity theoretical details.

We can always specify complex constructions in terms of abstract primitives that are known to satisfy certain functionality and security guarantees. In other words, we actually specify a template that can be instantiated by replacing abstract primitives with practical constructions. Therefore, the corresponding security proofs are free from implementation details and rely mostly on the elementary probability theory. Of course, the complexity theoretical problems do not disappear entirely. They re-emerge as soon as we try to establish whether a concrete realisation of an abstract primitive indeed satisfies the necessary security requirements or not. However, the latter is the main concern of primitive design and thus beyond our scope.

Such an approach leads to a hierarchy of cryptographic abstractions. At the lowest level of this hierarchy, we find many low level primitives that formalise our current beliefs and knowledge about computational complexity. For example, we can talk about abstract Diffie-Hellman groups and abstract sets of RSA composites that formalise certain complexity theoretical beliefs about the hardness of discrete logarithm and integer factorisation. Again, we must use the common methodology of primitive design to establish whether such abstract constructions can be implemented in practice. Primitives that realise some well-specified functionality, such as encryption or authentication, are often specified in terms of lower level primitives. For example, ElGamal and Rabin encryption schemes are built on top of Diffie-Hellman groups and RSA composites. The next abstraction level consists of lower level protocols that are designed and optimised for well-specified tasks, such as authentic message delivery or entity identification. These basic protocols can be combined in order to achieve more complicated tasks like online banking and digital rights management. The hierarchy of abstractions continues—we can combine these protocols to achieve even more complex tasks. Since there are so many different levels, it is essential to have a flexible formalism that makes it easy to change the levels of abstraction, otherwise we are either trapped in the ocean of unnecessary technical details or the viewpoint is too coarse for performance optimisations.

Still, many proofs are inherently complex, even if we choose a correct abstraction level, and occasionally we must increase the granularity to discover performance bottlenecks. In these cases, we must handle proofs that are tens, if not hundreds, of pages long. For obvious reasons, such proofs quickly become unmanageable, unless they are extremely well structured. Ideally, these proofs

should be automatically verifiable. Such a goal seems extremely hard to achieve, as cryptographic proofs are often based on complex reductions and thus require creative manipulation of probabilities. In Chapter 4, we explore this issue further and show that all proofs share the same meta-structure. Any proof can be formalised as a game tree, where each edge corresponds to an elementary proof step. Although each elementary proof step itself may rely on complex reasoning, it is easy to describe the related syntactical change in the game. More importantly, proof steps that lead to the same kind of syntactic changes also reuse the same proof template. Therefore, we can formalise such steps as reduction schemata. Notably, the total number of reduction schemata is rather limited, since each primitive introduces few elementary reduction schemata and other more complex reductions can be expressed as mixtures of those.

As reduction schemata abstract away complex technical details, the resulting proof skeleton is compact and easily verifiable. In fact, a person who applies reduction schemata does not have to know the exact technical details as long as he or she applies reduction schemata correctly. In short, derivation of security proofs is mostly an engineering task, whereas formalisation and systematisation of reduction schemata requires a deeper understanding of cryptography. We believe that such a representation of basic concepts brings us closer to computer-aided design of complex cryptographic primitives and protocols. The need for such design assistants becomes self-evident when we compare cryptography to other engineering disciplines, where hundreds or even thousands of general and special purpose design programs are routinely used by engineers. Although there are many experimental protocol analysers, most of them put too much effort on automatic derivation of proofs. The author believes that the latter is a secondary goal and the most critical part is a human readable representation of proofs and the ability to apply various reduction schemata without errors. Of course, the development of such a computer environment is a major undertaking and it may take several man-years to achieve significant progress.

## 1.2 INTRODUCTION TO CRYPTOGRAPHIC PROTOCOL DESIGN

A conceptual difference between cryptographic primitives and protocols lies in the security guarantees. Primitives are designed to preserve security in few well-defined scenarios, whereas protocols must resist different attacks aimed at different targets. Often, there are literally thousands and thousands of potentially relevant security goals and we cannot analyse them separately. Instead, we should compare different protocols  $\pi_\alpha$  and  $\pi_\beta$  for the same functionality. Beaver was the first to state the corresponding resilience principle [Bea91a]:

*If any adversary attacking  $\pi_\alpha$  cannot gain more information or wield more influence than when it attacks  $\pi_\beta$ , then  $\pi_\alpha$  is at least as secure and reliable—i.e. as resilient—as  $\pi_\beta$ .*

Evidently, a protocol  $\pi_\alpha$  attains a maximal security level if  $\pi_\alpha$  is as resilient as any other protocol  $\pi_\beta$  that correctly implements the same functionality. As a direct comparison with any other imaginable protocol  $\pi_\beta$  is infeasible in practice, we must use a methodological shortcut. Namely, we can easily define an ideal implementation  $\pi^\circ$  that is as resilient as any other protocol  $\pi_\beta$ . Since resilience

is transitive by definition, it is sufficient to compare only protocols  $\pi_\alpha$  and  $\pi^\circ$  to establish optimality. The corresponding real versus ideal world paradigm is a central methodological approach in cryptographic protocol design.

However, the exact nature of these security guarantees depends on many technical details. Most importantly, we must define comparability between attacks against real and ideal world implementations. In particular, note that participation in a protocol is economically justified if potential gains outweigh associated risks. Thus, we must assure that all relevant economical aspects are also captured by the ideal implementation, otherwise real and ideal implementations are not truly comparable. In Chapter 5, we study this problem thoroughly in the simplest stand-alone setting, where a protocol is executed in isolation and the adversary cannot use external information. Although the resulting security definitions are equivalent to the standard formalisations [Can00a, Gol04], our treatment also reveals underlying causes why the definitions must be formalised in such a way.

Again, note that an appropriate solution for a specific task is often a balanced mixture of cryptographic and non-cryptographic measures. Therefore, different applications utilise different cryptographic properties. In Chapter 5, we describe the most common security objectives. We start from the semi-honest model, where all participants follow the protocol, and finish with the malicious model, where the set of malicious participants can arbitrarily deviate from the protocol. Since we consider input-privacy and output-consistency in addition to complete security, we obtain four security models that capture the most essential security requirements. Although practical settings are often more complex, they are usually just intelligent variations of these basic models.

The biggest advantage of the stand-alone setting is conceptual simplicity, which makes it easy to formalise and analyse various security objectives. However, the corresponding security models explicitly assume that adversaries are isolated from external influences. The assumption rarely holds in practice, where protocols are commonly used as subroutines to achieve more complex goals. Hence, it is essential to know which computational contexts preserve the security of the protocol. First, if we know the corresponding usage restrictions, we can combine various protocols without unexpected consequences. Secondly, usage restrictions are actually security guarantees that describe when it is secure to execute a protocol. Since the whole computational context is often unknown to the participants, it is important to state usage restrictions in local terms. We emphasise that there is a natural trade-off between usage restrictions and the efficiency of a protocol. In other words, we cannot rank protocols only according to the usage restrictions, since protocols with the most liberal usage restrictions might have sub-optimal performance and unjustified deployment costs.

Although it is relatively straightforward to understand what a computational context is, the corresponding formalisation is cluttered with many tedious technical details. Hence, we try to present the complex computational model as modularly as possible in Chapter 7. We also study the true limitations of stand-alone security and establish the corresponding usage restrictions. Briefly, stand-alone security is sufficient for all computational contexts, as long as we are willing to assume that all external participants<sup>1</sup> are corrupted during the protocol. Al-

---

<sup>1</sup>More formally, external participants are parties that do not participate in the protocol.

though the result is a minor enhancement of classical sequential composability theorems [Ore87, Can00a], it bears a huge conceptual significance. Namely, it implies that stand-alone security is sufficient for all end-to-end applications that treat all external participants as enemies.

However, the corresponding usage restrictions are also quite limiting, since participants in a stand-alone secure protocol cannot perform any side-computations during the execution of the protocol. As a result, we obtain a centralised execution model that treats the whole network as a single virtual processor. Since network-wide synchronisation is often impossible or causes significant performance penalties, we need protocols with more liberal usage restrictions. Surprisingly, we can significantly relax usage restrictions for stand-alone secure protocols as soon as the correspondence between real and ideal world adversaries satisfies certain structural restrictions already suggested in early works [Bea91b, MR91b]. In fact, such protocols are called universally composable, as they preserve security in any reasonable context. Moreover, all-or-nothing results by Lindell [Lin04] indicate that a general class of protocols with simple and natural usage restrictions consists of protocols that are either secure only in the stand-alone model, or universally composable.

The structural restrictions mentioned above are difficult to fulfil. In fact, it is impossible to achieve universal composability when honest participants are in minority, i.e., malicious participants have more control over the protocol than the coalition of honest parties. Fortunately, there is a loophole that allows us to bypass the all-or-nothing nature of usage restrictions. Namely, we can often locate critical regions in a stand-alone protocol that must be executed in isolation, whereas the other regions are indeed universally composable. Consequently, our task is to minimise the duration and count of these critical regions without losing the overall efficiency. Chapter 7 covers also the corresponding three-stage design methodology and studies protocols with a trusted setup.

As a final detail, note that cryptographic protocol design borrows many concepts from hypothesis testing and thus it also inherits many limitations of classical statistics. In particular, classical statistics describes only average-case behaviour and cannot be used for analysing individual experiments. Analogously, classical cryptography describes only collective security properties and fails when we want to analyse the security of specific instances. In other terms, design and usage of cryptographic protection measures are two separate things. As an illustrative example, consider a nation-wide public-key infrastructure. A well-designed signature scheme assures that the number of weak signing keys and potential monetary risks are negligible for the organisation that issues keys for all individuals. However, the latter does not imply that a particular signing key is secure to use. In fact, we cannot even formalise the corresponding design goal by using classical statistics. The latter is a serious drawback as a key holder is only interested in the security level provided by his or her signing key. In Chapter 6, we show how to formalise such security goals by using the notion of subjective probability. While we acknowledge that such concepts and security goals are somewhat non-standard in cryptographic literature, the resulting formalism is mathematically sound and well-aligned with practice.

### 1.3 BENEFITS OF FINITE SET POLICY

There are many ways to obtain mathematically sound security definitions; exact quantification of security properties is only one of them. In particular, various asymptotic security models are widely used in theoretical cryptography, since these models are well aligned with complexity theory and also hide many tiresome technical details. Nevertheless, there are several compelling reasons why we should stick to exact quantification of security. First, connections between theoretical results and practical consequences are significantly weaker and we must address the limitations already raised by Yao [Yao82]:

*In this theory, the complexity of a computational problem is measured by the asymptotic behaviour of algorithm as the input length becomes large. To apply the theoretical result to input of a particular length, we tacitly assume that this length is large enough that the asymptotic results can be used.*

Secondly, asymptotic behaviour can be surprisingly brittle and counterintuitive. Misunderstanding of limiting processes has created countless errors and paradoxes throughout the history of mathematics. There is no reason to believe that cryptography or computer science are special in that respect and thus we should always follow the finite set policy [Jay03]:

*Apply the ordinary process of arithmetic and analysis only to expressions with a finite number of terms. Then, after the calculation is done, observe how the resulting finite expressions behave as the number of terms increases indefinitely.*

Less formally, we first formulate models with precise bounds and only then observe limiting behaviour. Indeed, violation of this simple principle has led to subtle errors or inconsistencies in cryptography. For example, aesthetic reasons urged theoreticians to consider also algorithms that run in expected polynomial time, see [Gol07] for further discussion. As a result, we have now at least four different notions of polynomial security, which are incompatible. To make things worse, several important results, such as composability theorems, do not hold for all of these models, see the first half of Chapter 6. At the same time, there exists basically one finite security model and we can safely apply a theorem as soon as all quantitative requirements of the theorem are satisfied.

Another notorious example is the definition of universal composability. Depending on subtle details in definitions, some important equivalence results either hold or not, see the discussion in [Can00b]. In fact, there are at least four different asymptotic definitions of universal composability [DKMR05], whereas there is a single finite security model for universal composability. All these results clearly demonstrate that skipping the finite modelling step is dangerous.

Finally, we remark that many asymptotic models use a complex limiting process over several variables to characterise many aspects in a single stroke. For example, a zero-knowledge proof can be characterised by the complexity of the statement to be proved and three other quantitative properties: correctness, soundness and zero-knowledge property. However, the corresponding asymptotic model is unidimensional and thus many details that are important in practice are erased by an inherent dimension reduction.

## 1.4 CONTRIBUTIONS OF THE AUTHOR

This thesis has a slightly non-standard structure. Most theses are either monographs or contain a brief summary and a listing of published articles, whereas this thesis consists of a monograph followed by a few selected articles. Such a choice was made mostly because the author felt compelled to first systematically present essential theoretical concepts and then use the articles as illustrations for the covered topics. The author believes that such a systematic approach is justified, since an engineering view on cryptography is currently underrepresented in mainstream literature. Although exact quantification of security properties is quite common in cryptographic primitive design, the corresponding formalisations are often very application specific. Moreover, all standard treatments of secure multi-party computations [MR91b, Can00a, Can00b, Gol04] are based on asymptotic settings and do not place results into a larger economical context. Hence, the need for systematical treatment of exact security is real and hopefully this thesis is a step to the right direction. Also, most of the topics covered in this thesis are closely connected with the problems that the author has faced in his research. However, we put the main emphasis on the methodological issues, since protocol design itself becomes a mundane engineering task as soon as methodological issues are resolved and design criteria are fixed.

As this thesis covers many essential results, it is somewhat difficult to separate the author's original contribution from prior art. Therefore, we list the most important novelties and their connections to the author's published articles. Conceptually, the most important novelty is an engineering viewpoint to cryptographic protocol design and the coherent use of exact security guarantees throughout this thesis. Differently from standard treatments, we use hypothesis testing as a main tool and design a primitive or a protocol only from abstract primitives that completely formalise all relevant security properties. Such an approach allows us to present proofs in a more systematic manner and use (semi-)formal methods to derive and verify security proofs. The corresponding formalism established in Chapter 4 is a novel addition to game-playing proofs popularised by Shoup and Rogaway, see for example [Sho04, BR06]. First, we demonstrate that some security proofs cannot be converted to game chains and show how to formalise these proofs by using game trees. Although such proofs are quite rare in primitive design, they are often required in the security analysis of protocols. In fact, the author experienced these limitations by analysing authentication protocols in [LN06] and the proof methodology presented in Section 4.4 is just a direct generalisation of the techniques used in [LN06]. Secondly, we go one step further than the standard game-playing proofs [Sho04, BR06] and show how to abstract away all technical details and express the skeleton of the proof via syntactic changes formalised as reduction schemata. The approach differs from classical security calculi, as the proof skeleton is informative enough to reconstruct all technical details, including the bounds on advantages and running times. The author has found the corresponding formal approach invaluable for deriving and verifying technically complex reductions such as [LP08].

Thinking in terms of reduction schemata forces us to generalise individual proof steps into parametric proof templates. The corresponding meta-level analysis often reveals analogies between proof steps that are formulated for completely different primitives, but use technically identical arguments. For exam-

ple, the author’s study of specific collision extraction techniques [KLL06, BL07] has shown that the corresponding security analysis deals with a certain combinatorial problem. Hence, the results obtained in [KLL06, BL07] are not specific to a certain primitive; but can be used in the contexts of hash functions, commitment schemes, proofs of knowledge and digital signatures. We acknowledge here that these results contain mostly technical improvements, but the generality and optimality of the obtained bounds has a definite engineering value.

Time-success and risk profiles introduced in Chapters 4 and 5 is another methodological advance that improves our understanding about cryptographic reductions. In many security proofs, we must choose between alternative reduction methods, which lead to incomparable security guarantees. Consequently, we need a methodology for comparing radically different reduction strategies. Although the problem caught the author’s attention already in the article [BL07], the issue was fully resolved during the preparation of the manuscript [KLL06] that lead to the current formalisation presented in Section 4.2.

Also, note that standard formalisations of secure multi-party computations just postulate the security criterion without explaining whether it also makes sense from an economical viewpoint. Now, considering the problem in terms of time-success and risk profiles provides an elementary explanation why these security definitions are meaningful in the economical context as well.

As the third methodological advance, we explicitly discuss intermediate security levels between security in the semi-honest model and complete security in the malicious model. Such models provide cost-efficient solutions for many practical settings. In particular, we have studied many privacy-preserving data mining tasks [GLLM04, LLM05, LLM06] that become practically intractable in the malicious setting, if we want to achieve complete security. Still, we can guarantee the privacy of inputs by using the techniques developed in the article [LL07]. Moreover, we have shown that one can detect cheating with a significantly lower overhead than is needed for complete security [LL06].

Another, more debatable but still conceptually novel enhancement by the author is a rigorous formalisation of subjective security. In Chapter 6, we define the corresponding notion and explain why classical security definitions are inadequate for certain settings. Moreover, we also explain why random oracle or generic group models provide only subjective security guarantees and why all standard arguments, such as [CGH04b], against these models are unconvincing.

A final and probably the most important methodological advance is a coherent and practice-oriented treatment of modular protocol design. Differently from the standard approach, we treat universal composability as a tool and not as the final aim. In particular, we present an efficient three-stage design methodology that significantly simplifies the construction of complex protocols without losing efficiency. The corresponding design process often leads to protocols that share the same trusted setup phase. As all classical composability theorems are inadequate in such settings, we must use more fine-grained methods. The corresponding methodology presented in Section 7.7 is a direct generalisation of proof methods used in the article [LL07], where we proved composability theorems for very specific two-round protocols with a shared key generation phase. Similar theoretical concerns emerge also in the articles [LN06, LP08] where protocols are proved to be secure in the common reference string model.

**On the choice of selected articles.** As explained above, the first part of this

thesis covers and generalises the most important methodological advances made by the author. Thus, we include only four articles [GLLM04, LN06, BL07, LL07] out of eight publications into this thesis. This does not mean that the remaining four publications [LLM05, BL06, LLM06, LP08] are technically less sound or otherwise inferior. Neither was the choice based on the author's impact on the articles. The author has made a significant contribution to all eight articles. The choice was made mainly on the basis that the selected articles illustrate certain theoretical aspects and methods more vividly.

Published articles cover three different sub-fields of protocol design: privacy-preserving data mining, time-stamping and message authentication. To be precise, we consider only such tasks of privacy-preserving data mining that are special instances of secure multi-party computation. In a nutshell, we assume that the data is split between several participants and they want to learn global patterns in the data without revealing more information than absolutely necessary. Many data mining algorithms are based mostly on linear vector and matrix operations. Therefore, private evaluation of scalar products is one the most important tasks in privacy-preserving data mining and many authors have used ad hoc methods to solve the problem. The first included article [GLLM04] shows that these ad hoc protocols are not only cryptographically insecure but can completely reveal the private input if used in practical applications.

We remark here that ad hoc methods are still widely used in privacy-preserving data mining, since cryptographic solutions are practically infeasible and people are actually willing to reveal more information during the execution than absolutely necessary. Consequently, the cryptographic models must be flexible enough to model various trade-offs between privacy and efficiency, in order to be useful. See Section 5.4 for the corresponding remarks. At the same time, the article [GLLM04] also emphasises that small leakages can accumulate and cause total disclosure and thus one must be extremely careful with such specifications. As a positive application of cryptographic techniques, we show that scalar products can be privately computed by using homomorphic encryption. Moreover, the follow-up articles [LLM05, LLM06, LL07] use homomorphic encryption to implement more complex privacy-preserving tasks.

The second included article [LL07] presents a general methodology for evaluating complex predicates by using homomorphic encryption. More precisely, we study a hybrid security model, where the owner of the private key is maliciously corrupted but all opponents are guaranteed to be semi-honest. We show that complex protocols can be methodically designed by using three elementary operations based on homomorphic encryption. In terms of Chapter 7, we establish that these protocols are universally composable even if they share the same trusted setup phase. We also show how to enhance the security of semi-honest protocols so that they become secure in the hybrid model described above. Although the article presents methodology for certain two-round protocols, the methodology can be generalised for more complex protocol classes.

The third included article [BL07] explores various properties of time-stamping schemes. Briefly, a time-stamping scheme is a multi-party protocol that implements an unalterable digital register of events. As such, time-stamping must be based on binding commitments, such as hash trees. However, state of the art results [BS04] indicate that not all binding commitments are suitable for time-stamping. More formally, there exists a binding commitment scheme that is not

provably secure for time-stamping if one considers only black-box reductions. In this article, we prove that bounded list and set commitment schemes are sufficient for time-stamping. Moreover, our security definitions are more precise and well aligned with the ideal versus real world paradigm. Secondly, the article also provides a nice example of how white-box reduction techniques can be built on top of constructive combinatorial arguments.

The last included article [LN06] contains practical protocols for manual data authentication and manual key agreement. The article was greatly motivated by the recent advances in wireless communication. Nowadays, many handheld devices, such as cell phones, music players, keyboards send and receive data over wireless links and are thus vulnerable to active attacks. Moreover, practical limitations prevent the use of pre-shared keys or public-key infrastructure. Hence, we can assure authenticity only with the help of human operators who transfer short messages from one device to another. The article presents protocols for message authentication and key agreement that achieve a maximal security level in such settings. Moreover, these protocols are now used in the Bluetooth and Wireless-USB standards, see the discussion in [LN06].

## 2 COMMON NOTATION AND BASIC CONCEPTS

History has repeatedly shown that notation is extremely important in mathematics. Hence, we try to make the notation as simple and mnemonic as possible. Sometimes we intentionally omit details that can be grasped from the context. On the other hand, many authors try to capture all details and leave no room for misinterpretation. Although such presentation is pleasingly rigorous, it is often tedious, not to say boring and incomprehensible. We, readers, tend to have a limited amount of memory and thus exposing all details often makes the notations so complex that it overshadows the concepts behind them.

Modern cryptography is built on various subfields of mathematics and computer science, such as computational algebra, complexity theory, probability theory and statistics. As this thesis considers only the design of cryptographic protocols, we can narrow the set of necessary mathematical concepts and build the theory on basic properties of algorithms, reductions, probability and hypothesis testing. Surprisingly, we can actually abstract away all complexity theoretical details and provide a clean conceptual design of protocols for many tasks without any knowledge of complexity theory. We emphasise here that we still need complexity theoretical conjectures to reason about security. However, these complexity theoretical conjectures come into play later, when we must construct cryptographic primitives used in the design.

Next, we briefly list basic notations and then discuss the central concepts, such as models of computation and probability, more thoroughly.

### 2.1 BASIC MATHEMATICAL CONCEPTS

**Functions and domains.** Cryptography deals mostly with discrete transformations; real-valued functions are used only in the security analysis. As usual the set of all finite bit strings is denoted by  $\{0, 1\}^*$ . The set of finite bit string sequences is denoted by  $\{0, 1\}^{**}$  and is identified with  $\{0, 1\}^*$ . Therefore, any uniquely determined discrete transformation can be viewed as a *deterministic function*  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  regardless of how many arguments the transformation has. Some discrete transformations are randomised. A *randomised function* is defined as  $f : \{0, 1\}^* \times \Omega \rightarrow \{0, 1\}^*$ , where  $\Omega$  is a corresponding sample space and the output  $f(x) := f(x; \omega)$  depends on the *random choice*  $\omega \in \Omega$ . A randomised function without explicit arguments  $X : \Omega \rightarrow \{0, 1\}^*$  is known as a *discrete random variable*. We always assume that the random choices for different functions are independent unless explicitly specified otherwise.

**Distributions and probability.** Randomised functions are inherently non-deterministic, as the random choice  $\omega$  is not known ahead. Probability theory provides qualitative methods for assessing the corresponding uncertainty. Usually, the probability is formalised according to the axiomatisation given by Kolmogorov [Bil95]. The latter requires a formal definition of measurable subsets  $\mathcal{F}(\Omega)$  and the corresponding probability measure  $\Pr[\cdot] : \mathcal{F}(\Omega) \rightarrow [0, 1]$ . Fortunately, we can omit these measure theoretical details and give a more elementary definition for discrete random variables and functions. Any *discrete*

distribution can be viewed as a random variable  $X : \Omega \rightarrow \{0, 1\}^*$  together with a sequence of non-negative numbers  $(p_x)_{x \in \{0, 1\}^*}$  such that  $p_x = 0$  for all  $x \notin X(\Omega)$  and

$$\sum_{x \in X(\Omega)} p_x = 1 . \quad (2.1)$$

Usually, one can only observe the output  $X(\omega)$  and not the underlying random choice  $\omega \in \Omega$ . Therefore, we can limit our attention to externally observable events  $X(\omega) \in \mathcal{A}$  where  $\mathcal{A} \subseteq \{0, 1\}^*$  and define the probability measure as

$$\Pr[\omega \leftarrow \Omega : X(\omega) \in \mathcal{A}] := \sum_{x \in \mathcal{A}} p_x . \quad (2.2)$$

The equations (2.1) and (2.2) assure that the corresponding probability measure is indeed well-defined and satisfies Kolmogorov's axioms.

To simplify notation, we use the following conventions. A shorthand  $x \leftarrow X$  denotes that  $x$  is chosen according to the distribution  $X$ . Since a randomised function  $f : \{0, 1\}^* \times \Omega \rightarrow \{0, 1\}^*$  defines a distribution  $F_z(\omega) = f(z; \omega)$  for every input  $z$ , we can use analogous shorthand  $y \leftarrow f(z)$  to denote that  $y$  is chosen according to the distribution  $F_z$ . Finally, let  $\mathcal{A}$  be a finite subset of  $\{0, 1\}^*$  consisting of  $n$  elements. Then a shorthand  $x \leftarrow \mathcal{A}$  denotes a uniform distribution over  $\mathcal{A}$ , i.e.,  $\Pr[x \leftarrow \mathcal{A} : x = a] = \frac{1}{n}$  for all  $a \in \mathcal{A}$ .

**Asymptotics and time complexity.** Cryptographic proofs often contain many lower or upper bounds. In many cases, we are only interested in the asymptotic behaviour of these bounds. Then so called Landau notation can significantly simplify the derivation and the form of corresponding bounds.

Let  $f$  and  $g$  be non-negative functions that are defined over the positive real numbers. Then  $f(x) = O(g(x))$  denotes that there exists constants  $c, x_0 > 0$  such that  $f(x) \leq c \cdot g(x)$  for all  $x > x_0$ . Similarly,  $f(x) = \Omega(g(x))$  denotes that there exists constants  $c, x_0 > 0$  such that  $c \cdot g(x) \leq f(x)$  for all  $x > x_0$ . Now  $f(x) = \Theta(g(x))$  stands for  $f(x) = O(g(x))$  and  $f(x) = \Omega(g(x))$ . Finally, symbols  $o(\cdot)$  and  $\omega(\cdot)$  denote differences in the growth rates:  $f(x) = o(g(x))$  if  $f(x)/g(x) \rightarrow 0$  and  $f(x) = \omega(g(x))$  if  $f(x)/g(x) \rightarrow \infty$ , as  $x \rightarrow \infty$ .

Asymptotic behaviour can be studied in two different contexts. First, we can investigate how the exact security bounds of the basic primitives influence the overall security of a particular construction. For example, we can characterise how a discovery of a new more efficient factoring algorithm influences the overall security if a construction is based on 1024-bit primes. Secondly, we can consider construction families that are indexed by a *security parameter*  $\mathbb{k}$  and observe asymptotic behaviour of security guarantees as a function of  $\mathbb{k}$ . Such an approach helps us to choose optimal constructions for specific security levels. In particular, we can consider various complexity classes for different time bounds. A time bound  $f(\mathbb{k})$  is *polynomial* if  $f(\mathbb{k}) = O(\mathbb{k}^c)$  for  $c > 0$  and *polylogarithmic* if  $f(\mathbb{k}) = O(\log(\mathbb{k})^c)$  for  $c > 0$ . The corresponding shorthands are  $f \in \mathbf{poly}(\mathbb{k})$  and  $f \in \mathbf{polylog}(\mathbb{k})$ . Similarly, a function  $\varepsilon(\mathbb{k})$  is *asymptotically negligible* denoted by  $\varepsilon \in \mathbf{negl}(\mathbb{k})$  if it decreases faster than a reciprocal of any polynomial, i.e.,  $\varepsilon(\mathbb{k}) \in o(\mathbb{k}^{-c})$  for all  $c > 0$ . We deliberately emphasise the asymptotic nature of this notion and reserve the term *negligible* for values that are insignificant in practice, say less than  $2^{-80}$ . Similarly, *overwhelming probability* is a constant sufficiently close to 1 and not an asymptotic notion.

## 2.2 DIFFERENT INTERPRETATIONS OF PROBABILITY

Most of the cryptographic security definitions are explicitly based on the concepts of hypothesis testing. Therefore, cryptography also inherits many problems and shortcomings from statistics. In particular, we must interpret probabilities. Interpretation of probabilities is by no means an easy and unambiguous task as there are at least five different interpretations. Basic differences of interpretations concern objectivity or subjectivity of probabilities.

Imagine that you throw an unbiased coin. Then objective probability describes the potential of seeing heads and subjective probability your rational opinion about the odds of seeing heads. In particular, the objective probability disappears when the coin has landed even if you do not see it. The subjective probability can exist also after the landing provided that you do not see the outcome. In other words, objective probability describes a potential that realises in the future, whereas subjective probability can handle completed events with unknown outcomes. In that sense, objective probability describes always collective properties—characteristics of possible future events—and subjective probability individual properties—characteristics of a fixed event. This conflict between individual and collective properties is an unresolved theme in statistics.

Frequentists believe that probability is an *inherently collective* property of distributions that manifests itself as a limiting relative frequency of independent repeated trials. This line of thought was initiated by Cournot, Venn and Ellis in the middle of the 19<sup>th</sup> century. These ideas were extended further by von Mises in 1919 who explicitly stated that probability is a property of infinite sequences (*collectives*) that satisfy certain conditions. In particular, all “admissibly selected” sub-sequences must have the same limiting frequency as the sequence itself, i.e., no gambler can devise a winning strategy with fixed bets. Such a strong form of frequentism was refuted in 1937 by the contemporary mathematicians, as the theory had several “inconsistencies” and Kolmogorov had provided a better alternative in 1933, see [Fie92, vL96, Bin00, Sti86, LV97] for further comments and historical remarks. Such development boosted the popularity of mathematical statistics as a more liberal branch of frequentism and it became a prevalent school of thought. Starting from seminal works of Pearsons, Neyman and Fisher the main emphasis shifted to the design inference procedures (algorithms) that behave well on average over the distributions.

To illustrate the tension between objective and subjective probability in the context of cryptography, consider the security of a cryptosystem. Recall that a cryptosystem is a collection of sub-instances indexed by a randomly chosen secret key. In particular, we can fix an optimal attack algorithm and ask how well the cryptosystem as a collection of sub-instances survives the attack on average. Such a frequentistic viewpoint is appropriate in the design and evaluation of cryptosystems. Assume that an attack succeeds with negligible probability on average. Then by the law of large numbers, we expect that only a tiny, not to say non-existent, fraction of users to suffer from the attack. However, if we take the position of a user then the tables are turned. A user could not care less about the overall security as long as his or her particular sub-instance is immune against the attacks. Such a problem is *ill-posed* in the frequentistic setting: the secret key has already been chosen, we just do not know if the *particular* key is easier or harder to break than the average key.

To answer the questions about individual instances, it is *inevitable* to accept the notion of subjective probability. This line of thought was started by Bayes and was later rediscovered by Laplace. According to them, the probability is a quantitative measure of uncertainty that can be rationally updated by observing empirical data. During the early twenties and thirties of the 20<sup>th</sup> century the notion subjective probability was intensively studied by economists and mathematicians. Keynes and Ramsey were the first to propose formal theories that treated probability as a measure of uncertainty. The process was later completed by de Finetti who showed in 1937 that any rationally behaving entity must assign prices to bets that confirm Kolmogorov's axioms and satisfy Bayes' theorem. These results were later reproved by Savage in 1954 and Cox in 1961 under different but natural assumptions about rational behaviour. We refer to [Fie05] for an excellent treatment of historical developments and to [Jay03, Jef04, BT92] for a thorough treatment of subjective probability.

Essentially, there are two schools of Bayesianism. In statistics the main goal is objectivity and thus the objective Bayesianism tries to minimise the impact of the subjectivity and make the end results universally acceptable. The subjective Bayesianism makes no efforts to achieve objectivity and is more common in economics and game theory, where the buyers or players may indeed have biased but still rational preferences. Since all individual decisions about a particular choice of cryptographic primitives *can* and *should* be treated in the framework of economical game theory, we follow the school of subjective Bayesianism.

Finally, we emphasise that both interpretations of probabilities and their corresponding extensions in cryptography are equally important. The objective treatment is a natural choice when we assess the general design of cryptographic solutions, whereas one cannot avoid subjectivity if we want to make theoretically well-founded user level decisions, see the thorough discussion in Chapter 6.

## 2.3 BASIC PROPERTIES OF RANDOM VARIABLES

**Independence and product space.** Two random variables  $X : \Omega_1 \rightarrow \{0, 1\}^*$  and  $Y : \Omega_2 \rightarrow \{0, 1\}^*$  are independent if the corresponding random choices are independent. Formally, we must form a joint sample space  $\Omega = \Omega_1 \times \Omega_2$  with random variables  $X(\omega_1, \omega_2) := X(\omega_1)$  and  $Y(\omega_1, \omega_2) := Y(\omega_2)$  such that

$$\Pr [X \in \mathcal{A} \wedge Y \in \mathcal{B}] := \Pr [\omega_1 \leftarrow \Omega_1 : X \in \mathcal{A}] \cdot \Pr [\omega_2 \leftarrow \Omega_2 : Y \in \mathcal{B}] . \quad (2.3)$$

Alternatively, we can postulate that two random variables  $X$  and  $Y$  defined over the same sample space  $\Omega$  are independent if for all events  $\mathcal{A}$  and  $\mathcal{B}$

$$\Pr [\omega \leftarrow \Omega : X \in \mathcal{A} \wedge Y \in \mathcal{B}] = \Pr [\omega \leftarrow \Omega : X \in \mathcal{A}] \cdot \Pr [\omega \leftarrow \Omega : Y \in \mathcal{B}] . \quad (2.4)$$

**Conditional probability.** Let random variables  $X = X(\omega)$  and  $Y = Y(\omega)$  be defined over the same sample space  $\Omega$ . Then the conditional probability of the event that  $X \in \mathcal{A}$  given that  $Y \in \mathcal{B}$  is defined as

$$\Pr [X \in \mathcal{A} | Y \in \mathcal{B}] := \frac{\Pr [\omega \leftarrow \Omega : X(\omega) \in \mathcal{A} \wedge Y(\omega) \in \mathcal{B}]}{\Pr [\omega \leftarrow \Omega : Y(\omega) \in \mathcal{B}]} \quad (2.5)$$

provided that the denominator is not zero. In the Bayesian framework, a shorthand  $\Pr[X \in \mathcal{A} | Y \in \mathcal{B}]$  quantifies a subjective belief that  $X \in \mathcal{A}$  when  $Y \in \mathcal{B}$  is known to hold and *a priori* it is not clear that the equation (2.5) must hold. Nevertheless, it is possible to prove that the famous Bayes' theorem

$$\Pr[X \in \mathcal{A} \wedge Y \in \mathcal{B}] = \Pr[X \in \mathcal{A} | Y \in \mathcal{B}] \cdot \Pr[Y \in \mathcal{B}] \quad (2.6)$$

and all other Kolmogorov axioms still hold for any *rational belief system*, see the handbooks [Jay03, Jef04] for further details. In other words, all manipulation rules are formally the same for objective and subjective probabilities.

**Statistical distance.** Let  $X$  and  $Y$  be random variables. Then statistical distance (also known as statistical difference) between  $X$  and  $Y$  is defined as

$$\text{sd}(X, Y) := \sup_{\mathcal{A} \subseteq \{0,1\}^*} |\Pr[X \in \mathcal{A}] - \Pr[Y \in \mathcal{A}]|, \quad (2.7)$$

where the probabilities are computed over the corresponding distributions. It is straightforward to prove that statistical difference has an alternative form<sup>1</sup>

$$\text{sd}(X, Y) = \frac{1}{2} \cdot \sum_{x \in \{0,1\}^*} |\Pr[X = x] - \Pr[Y = x]| \quad (2.8)$$

and thus can be viewed as  $\ell_1$ -norm. Hence, it also satisfies the triangle inequality

$$\text{sd}(X, Z) \leq \text{sd}(X, Y) + \text{sd}(Y, Z). \quad (2.9)$$

Intricate connections between statistical distance and hypothesis testing and their extensions to computationally bounded cases are discussed in Chapter 3.

**Random tape model.** Many cryptographic models use coin-flipping as the only source of randomness, i.e., the probability space is  $\Omega_* = \{0, 1\}^*$  where each consecutive binary digit  $\omega_i \in \{0, 1\}$  is generated by flipping a fair coin. Note that it is fairly straightforward to show that any discrete distribution can be constructed in such a way. That is, for any sequence of non-negative numbers  $(p_x)_{x \in \{0,1\}^*}$  that satisfy the equation (2.1) there exists  $X : \Omega_* \rightarrow \{0, 1\}^*$  such that  $\Pr[X = x] = p_x$ . Moreover, the random variable  $X$  can be approximated with arbitrary precision using enough coin-flips. More formally, we can define a sequence of random variables  $X_n : \Omega_* \rightarrow \{0, 1\}^*$  such that  $X_n(\omega)$  depends only on the first  $n$  coin-flips  $(\omega_1, \dots, \omega_n)$  and  $\text{sd}(X_n, X) \rightarrow 0$ , as  $n \rightarrow \infty$ .

Surprisingly, a similar result holds in general. The Skorohod Representation Theorem assures that any probability distribution on an arbitrary set  $\mathcal{X}$  can be represented as a random variable  $X : \Omega_* \rightarrow \mathcal{X}$ . See the handbook [Bil95] for further details. Such a result is philosophically very pleasing, as it implies that the random tape model can still adequately represent reality.

---

<sup>1</sup>The sum on the right is absolutely convergent, since all terms are non-negative and the sum can be upper bounded by 2.

## 2.4 DIFFERENT FORMAL MODELS OF COMPUTATION

Computability is another central concept in cryptography, which is essentially a science of secure communication and computation. The notion of computability emerged in the first half of 20<sup>th</sup> century during the systematisation and re-evaluation of the foundations of mathematics. Initially, the emphasis was on the limitations of formal proof systems but later the attention turned to fully automatic derivation of numeric results. The remarkable breakthrough came in 1936 when Alonzo Church and Alan Turing independently showed that it is impossible to algorithmically decide whether a certain claim holds in arithmetic or not [Chu36, Tur37]. Both authors postulated that a rather simple model can be used to describe any fully mechanistic computational processes.

These postulates were quickly accepted and the corresponding thesis explicitly stated by Stephen Kleene [Kle43]. The Church-Turing thesis states that all “imaginable” algorithmic computations can be specified by Turing machines. Of course, the Church-Turing thesis can never be proved, since the notion of “imaginable” algorithmic computations is inherently vague.

We emphasise that the Turing machine is only a mathematical abstraction that allows us to formally model some aspects of modern computational devices. In the light of all known and foreseeable technologies, it is impossible to actually construct a Turing machine. The problem lies in the fact that Turing machine is an infinite object, whereas the observable universe is finite.<sup>2</sup> Briefly, a Turing machine is only a simple model of a modern computing device and not the other way around. Moreover, in many situations Random Access Machines reflect reality more precisely than Turing machines.

**Turing Machines.** A Turing machine can be viewed as a *finite automaton* with access to an infinite stack of memory cells. The stack is usually modelled by a one-sided infinite tape and a moving head that can read or write symbols underneath it. The movement of the head is controlled by an automaton with a finite number of states and a finite set of transition rules. A transition may depend on the symbol read by the head and it can specify a replacement symbol. However, the controller must be deterministic, i.e., there can be only a single transition rule for each state and a symbol underneath the head. A more formal treatment of Turing machines is given in textbooks [Pap93, HMU00].

Nevertheless, some peculiarities of Turing machines are worth stressing. By common convention, the *running time* of a Turing machine is just the number of transitions made by the controller before halting. In other words, the speed of an elementary read-write operation is independent of the size of finite controller. Therefore, for any predicate  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  there exist a Turing machine that outputs the correct answer for any input  $x \in \{0, 1\}^n$  in  $n$  steps. Basically, we can hardwire the binary search for the  $2^n$  element truth table. The corresponding  $2^{n+1}$  state finite controller finds the right entry in the truth table

---

<sup>2</sup>Astrophysicist have estimated that the mass of the Universe ranges from  $10^{53}$ – $10^{60}$  kg. Even if the Universe is infinite, then our current knowledge indicates that mankind can observe only a finite portion of the Universe during a finite time interval. Therefore, all modern computing devices are actually finite automatons whether we like it or not.

Another justification comes from the fundamental von Neumann-Landauer limit [Lan61]: any logically irreversible bit operation leads to unavoidable energy dissipation. Hence, only a finite number of logically irreversible operations can be carried out.

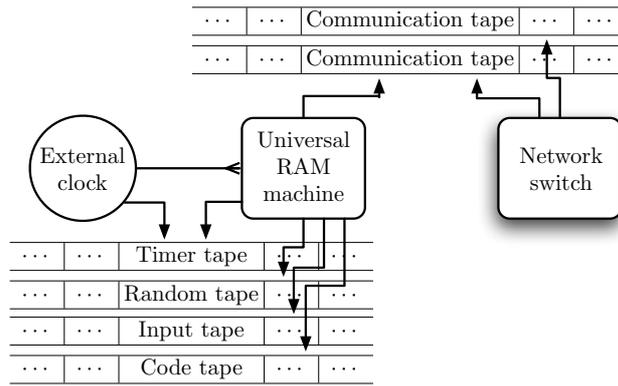


Figure 2.1: A schematic model of a computational node in a network

with  $n$  steps and outputs the corresponding result. More generally, any function  $f : \{0, 1\}^n \rightarrow \{0, 1\}^m$  can be computed in  $n + m$  steps.

To resolve the issue, we fix a universal Turing machine and require that any algorithm must be specified as a read-only bit-string (*program code*) on the special input tape. Such a model coincides with the *non-uniform complexity model* where the Turing machine is fixed but one can feed a special advice string that depends only on the input size  $n$ . Alternatively, we can directly penalise the Turing machine based on the size of the hardwired program. Interestingly enough, the latter leads to *circuit complexity*, as the computational logic can be completely encapsulated in the controller design. Although circuit complexity provides a better insight into how well an algorithm can be parallelised, the corresponding formalism is also much more difficult to understand.

**Random Access Machines.** Although Turing machines have no limits on the amount of accessible memory, it takes time to move the head to a random location. Often such linear memory access cost is not justified, since modern computing devices provide a large chunk of volatile memory that can be randomly accessed in a constant time. Similarly, read and write operations of modern hard disks can be considered location independent. Consequently, if a program has a small enough memory footprint, then a *Random Access Machine* (RAM machine) models computations more precisely. Recall that a RAM machine consists of a finite controller (*processor*) that can manipulate an infinite array of registers by sequentially processing a finite hardwired program. Informally, a RAM machine is just a program in some assembler dialect, i.e., it can do basic arithmetic operations with registers and conditional jumps. More formal definitions of RAM machines together with comparison to other computational models can be found in textbooks [Smi94, WW01, HMU00]. Similarly to Turing machines, we fix a single universal RAM machine to avoid an artificial speed up that is caused by hardwired binary search techniques.

**Standardised interface.** Although the computing process itself is handled differently by universal Turing and RAM machines, it is still possible to standardise how these *computing nodes* interact with the surrounding environment. Briefly, only three types of external events can influence computations: random choices, messages from other computing devices and timer interrupts. In the following, we gradually specify all these aspects of interaction, see Fig. 2.1.

**Randomised computations.** A computing node has two one-sided read-only input tapes: one for program code and one for randomness  $\omega \in \Omega_*$ . It also has a one-sided read-write tape for input and output. Given a code of  $\mathcal{A}$ , an input  $x$  and a random tape  $\omega$ , the interpreter  $\mathcal{U}$  starts to execute code. The computation ends successfully if the input is replaced by the output  $\mathcal{A}(x; \omega)$  and  $\mathcal{U}$  halts in the canonical end configuration. Otherwise, the output of  $\mathcal{A}$  is denoted by  $\perp$ . An algorithm  $\mathcal{A}$  is *deterministic* if no cells from the random tape  $\omega$  are read. A *randomised algorithm*  $\mathcal{A}$  can access a prefix of  $\omega$  proportional to running time. Hence, we can talk about output distribution of  $\mathcal{A}$  and the distribution of running time  $t(\omega)$  together with all statistical estimators.

**Interactive aspects of computing.** Besides classical input tapes a computing node has a pair of read-write tapes: one for communication and the other for timer services. The communication tape is connected to a *network switch*. A network switch is a dedicated machine that delivers messages between different nodes. More formally, a network switch has access to all communication tapes and whenever a new message is output by a node, it determines the recipient, copies the messages to the communication tape of the recipient and sends a notification message to the sender after that. Obviously, a certain communication protocol between nodes and network switch must be fixed, or otherwise some messages may get lost or infinitely delayed. Let us neglect the exact details and just postulate properties of a correct implementation: (a) no messages are dropped, inserted or modified; (b) each node has a physical network address; (c) a node always waits for a notification message that indicates a successful transfer; (d) messages are transferred in the same order as they are created.

Note that the actual time needed to copy messages is linear in the size of the message, whereas the network delay depends on the exact configuration and behaviour of the network. Finally, reliable point-to-point networking is only a stepping stone: more complex network topologies such as non-reliable and asynchronous communication can be built on top of the basic network layer.

**Complexity theoretical setting.** Turing and RAM machines model the timing aspects of computations differently, as they use different models of memory. More precisely, a random memory access takes  $\Theta(n)$  elementary operations for Turing machines and  $\Theta(1)$  operations for RAM machines, where  $n$  is the size of the memory bank. Neither of these estimates is truly accurate, as we can arrange memory cells into a three dimensional lattice. Thus, the maximal physical distance between two memory cells is  $\Theta(\sqrt[3]{n})$  and the corresponding access time is  $o(n)$ . Consequently, Turing machines overestimate and RAM machines underestimate the actual time needed to compute the outcome.

These time differences are irrelevant in the classical complexity theoretical setting that neglects polynomial factors. Although such an approach hides many tedious details, the resulting estimates are often too crude in practice. For example, algorithms with a quadratic asymptotic complexity are considered intractable in the context of large numerical computations, whereas algorithms with worst-case exponential complexity are successfully used in compilers. In reality, the tractability of an algorithm depends on the number of required elementary steps. Moreover, the effective time bound depends on the context. It can range from  $10$ – $10^{18}$  elementary operations depending whether it is a mission

critical real-time software or an earthquake prediction algorithm.

Secondly, setting up a correct asymptotic feasibility definition is not a straightforward task at all, since there are so many choices. Often these choices are irrelevant and lead to equivalent descriptions, whereas sometimes they make a big difference, see the discussion in manuscripts [Can00b, Gol07]. To avoid such surprises, we persistently operate with exact time bounds and consider asymptotic behaviour whenever complexity theoretic results are needed.

**Exact time complexity.** Unfortunately, the definition of a Turing machine is not robust if we consider exact time complexity. Namely, the running times of different algorithms depend on whether a Turing machine has zero, one or more than one internal working tape, see the discussion and further references in [LV97, p. 432–444]. Although the difference in running times can be at most quadratic, it is still very noticeable in practice. The same effect emerges also when the computations are divided between several Turing machines. Then the whole network has more working tapes at its disposal than a single Turing machine. Consequently, simulating an interactive computation on a single Turing machine takes significantly more time steps than the total running time of all the nodes in the network. Hence, the running time of an attack depends on whether an attacker controls a single or a cluster of Turing machines. Of course, the overhead is only polynomial in the total running time it but makes security analysis more complex. Hence, we state security results only for RAM machines, as they are more robust in terms of exact time complexity.

Let  $\mathcal{A}$  be an *algorithm*, that is, a finite program code for the universal RAM machine. Then the running time of  $\mathcal{A}$  is counted as follows. First, the program code of  $\mathcal{A}$  is read command by command to the main memory. Next, the program is interpreted according to a fixed instruction set such as [CR72, p. 75]. Each basic operation should require time at least proportional to the bit-length of the corresponding operands. Otherwise, we get paradoxical results such as a factoring algorithm that works in linear time in the input size [Sha79]. Now the exact time complexity of an algorithm can be defined either in a strict or liberal sense. An algorithm  $\mathcal{A}$  is a *t-time algorithm in the strict sense* if  $\mathcal{A}$  always halts after  $t$  elementary steps regardless of inputs and random coins. In reality, computations are often halted based on an external clock, i.e., the output of  $\mathcal{A}$  is set to  $\perp$  when the time limit  $t$  is exceeded. The latter gives a rise to the liberal definition of *t-time computations* that is used throughout the thesis.

Obviously, the definitions can be generalised to Turing machines but then one must explicitly fix the number of Turing machines controlled by an attacker and the number of working tapes inside a Turing machine.

**Time-bounded subroutines.** Most of the cryptographic proofs are based on black-box reductions, where an algorithm  $\mathcal{B}$  calls an algorithm  $\mathcal{A}$  as a subroutine to complete its task. Now the liberal definition of *t-time computations* causes a slight trouble. Imagine that a *t-time routine*  $\mathcal{A}$  succeeds with probability  $\frac{1}{2}$  and otherwise stays in the infinite loop. Then  $\mathcal{B}$  that calls  $\mathcal{A}$  must stop the subroutine  $\mathcal{A}$  after  $t$  steps, or otherwise  $\mathcal{B}$  also stays in the infinite loop. Obviously, such an external clock can be included into  $\mathcal{B}$  with the cost of polynomial slowdown. However, such slowdown is highly undesirable in our setting. Thus, we include timers explicitly into the computational model. Note that algorithms can use many timers, since subroutines can also call subroutines with timers.

For clarity, we use an infinite timer model, where an algorithm can set potentially unbounded number of timers. If a timer is fired then the algorithm is stopped and an interrupt handler is called. More formally, an algorithm  $\mathcal{B}$  can always write an address  $\mathcal{L}$  of an interrupt handler and a time bound  $t$  on the dedicated timer tape. Given  $\mathcal{L}$  and  $t$ , the timer decreases  $t$  until it reaches zero. Then the execution is continued from the  $\mathcal{L}^{\text{th}}$  command of  $\mathcal{B}$ .

Even more formally, an *external clock* is a dedicated machine with a single timer  $t$ , an address register  $\mathcal{L}$  and a potentially infinite stack containing time difference and address pairs  $(\Delta t_i, \mathcal{L}_i)$ . The external clock is coupled with the interpreter and each elementary step decreases the value of  $t$  until it reaches zero. If the timer value reaches zero, then the interpreter seeks the  $\mathcal{L}^{\text{th}}$  command from the code and continues execution from it. Next, the timer and the address registers are updated by the topmost pair  $(\Delta t_i, \mathcal{L}_i)$  in the stack. If the stack is empty then the external clock waits for an incoming query  $(t_i, \mathcal{L}_i)$  to update the timer and address registers. If a query  $(t_i, \mathcal{L}_i)$  arrives and the timer is already activated, then the clock computes  $\Delta t = t - t_i$ . If  $\Delta t \leq 0$  then the query is discarded, otherwise the values  $(\Delta t, \mathcal{L})$  are pushed on top of the stack and registers  $(t, \mathcal{L})$  are replaced by  $(t_i, \mathcal{L}_i)$ . As a final detail, note that computation of  $\Delta t$  must take into account the time needed to do all computations connected to stack updates to get timer interrupts at precise time moments.

The discussion above also shows that the infinite timer model is just a convenience and adversaries cannot use the timers for speeding up their computations. Indeed, the use of timers cannot cause a significant speedup, as the external clock can be realised by a single Turing or RAM machine.

### 3 HYPOTHESIS TESTING

One of the main objectives in cryptography is confidentiality. Throughout history various encryption schemes have been used to scramble communication so that eavesdroppers cannot reconstruct the original messages. However, sometimes even partial message recovery can be as disastrous as full disclosure. Generally, secrecy is preserved only if an adversary cannot reject or accept non-trivial hypotheses about secret inputs given access to public outputs. Thus, we must formalise secrecy goals in the framework of hypothesis testing.

More formally, let  $s$  be a secret input and  $f(s)$  the corresponding public output  $x$  where  $f$  is a publicly known randomised function. Then we can state at least one trivial hypothesis  $s \in \{0, 1\}^*$  and many non-trivial hypotheses such as  $s \in \{1, 4, 7\}$ . We say that  $\mathcal{H}$  is a *simple hypothesis* if it uniquely determines the output distribution, i.e., probabilities  $\Pr[x|\mathcal{H}]$  are well-defined for any  $x \in \{0, 1\}^*$ . For obvious reasons, any hypothesis can always be represented as a union of simple hypotheses. Therefore, we start with a question of how to choose between two simple hypotheses  $\mathcal{H}_0$  or  $\mathcal{H}_1$  and then gradually extend the approach to consider more and more complex settings.

#### 3.1 SIMPLE HYPOTHESIS TESTING

**Statistical tests.** Assume that an adversary must choose one of two simple hypotheses  $\mathcal{H}_0$  or  $\mathcal{H}_1$ . A *statistical test*  $\mathcal{A} : \{0, 1\}^* \rightarrow \{0, 1\}$  is a randomised predicate that specifies whether to accept or reject the *null hypothesis*  $\mathcal{H}_0$ . The rejection probability of  $\mathcal{H}_0$  is often denoted by  $\phi(x) = \Pr[\mathcal{A}(x) = 1]$ . A statistical test  $\mathcal{A}$  can make two types of errors: *false negatives* and *false positives*.<sup>1</sup> The ratio of false negatives is commonly denoted by  $\alpha(\mathcal{A})$  and the ratio of false positives is denoted by  $\beta(\mathcal{A})$ . In other words

$$\alpha(\mathcal{A}) = \Pr[\mathcal{A}(x) = 1|\mathcal{H}_0] \quad , \quad (3.1)$$

$$\beta(\mathcal{A}) = \Pr[\mathcal{A}(x) = 0|\mathcal{H}_1] \quad . \quad (3.2)$$

A common task in statistics is to find a strategy  $\mathcal{A}$  that minimises the ratio of false positives  $\beta(\mathcal{A})$  given a bound on false negatives  $\alpha(\mathcal{A}) \leq \alpha_0$ . A statistical test  $\mathcal{A}_*$  is *optimal w.r.t. the bound  $\alpha_0$  on false negatives* if

$$\beta(\mathcal{A}_*) = \inf \{ \beta(\mathcal{A}) : \alpha(\mathcal{A}) \leq \alpha_0 \} \quad . \quad (3.3)$$

The most obvious way to choose between hypotheses  $\mathcal{H}_0$  and  $\mathcal{H}_1$  is based on the likelihood of outputs. This test is known as the *likelihood ratio test*:

$$\phi_{\text{lrt}}(x) = \begin{cases} 1 & \text{if } \Pr[x|\mathcal{H}_0] < \eta \cdot \Pr[x|\mathcal{H}_1] \quad , \\ \rho & \text{if } \Pr[x|\mathcal{H}_0] = \eta \cdot \Pr[x|\mathcal{H}_1] \quad , \\ 0 & \text{if } \Pr[x|\mathcal{H}_0] > \eta \cdot \Pr[x|\mathcal{H}_1] \quad . \end{cases} \quad (3.4)$$

Observe that  $\alpha(\phi_{\text{lrt}})$  decreases monotonically when  $\eta \in [0, \infty)$  is decreased and  $\rho \in [0, 1]$  only determines how the ties are handled, see Fig. 3.1. Hence, for any

<sup>1</sup>Due to historical reasons, statisticians use terms Type I and Type II Errors instead.

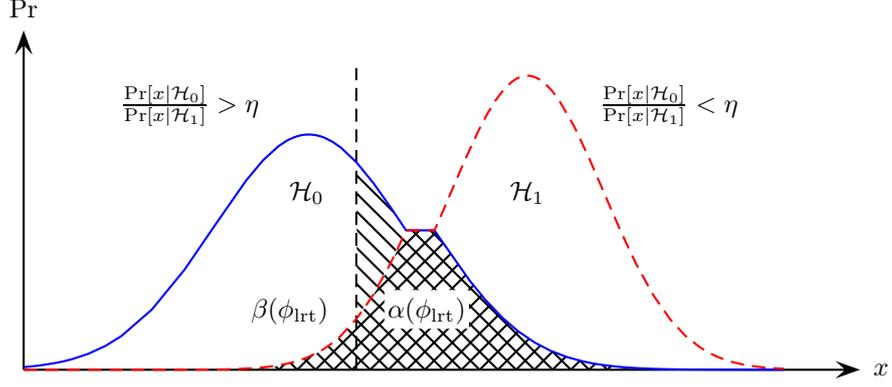


Figure 3.1: Likelihood ratio test. Decreasing  $\eta$  moves the decision border to the right and thus decreases  $\alpha(\phi_{\text{lrt}})$  and increases  $\beta(\phi_{\text{lrt}})$ . The aggregate error  $\gamma(\phi_{\text{lrt}})$  is minimised when it coincides with the double crossed area.

$\alpha_0$  there exists a likelihood ratio test that achieves  $\alpha(\phi_{\text{lrt}}) = \alpha_0$ . Moreover, the famous Neyman-Pearson Theorem states that the likelihood ratio test is optimal and every optimal test is functionally indistinguishable from the corresponding likelihood ratio test. See statistical handbooks such as [CB02] for more details.

Depending on the task at hand, the adversary can either reduce the ratio of false positives  $\alpha(\mathcal{A})$  by increasing the ratio of false negatives  $\beta(\mathcal{A})$  or vice versa. Although different tests can provide different trade-offs, we can still establish lower bounds for the *aggregate error*  $\gamma(\mathcal{A}) = \alpha(\mathcal{A}) + \beta(\mathcal{A})$ . Indeed, define a statistical distance between  $\mathcal{H}_0$  and  $\mathcal{H}_1$  w.r.t. the output  $x$  as

$$\text{sd}_x(\mathcal{H}_0, \mathcal{H}_1) = \sup_{\mathcal{A}} |\Pr[\mathcal{A}(x) = 1 | \mathcal{H}_0] - \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1]| \quad . \quad (3.5)$$

Then  $\gamma(\mathcal{A})$  of a statistical test  $\mathcal{A}$  is lower bounded as follows:

$$\gamma(\mathcal{A}) = 1 + \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_0] - \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1] \geq 1 - \text{sd}_x(\mathcal{H}_0, \mathcal{H}_1) \quad . \quad (3.6)$$

The inequality (3.6) also provides nice geometrical proof that the formula (2.8) indeed computes the statistical distance. By the construction

$$\gamma(\phi_{\text{lrt}}) \geq \sum_{x \in \{0,1\}^*} \min \{ \Pr[x | \mathcal{H}_0], \Pr[x | \mathcal{H}_1] \} \quad (3.7)$$

where the equality holds for parameters  $\eta = 1$  and  $\rho = \frac{1}{2}$ . Again, by geometrical considerations the right hand side of (3.7) can be expressed as the formula (2.8). Since the likelihood test is optimal, the formulae (2.7) and (2.8) must be equal.

**Computational tests.** Although Fig. 3.1 correctly illustrates properties of optimal statistical tests, it is also slightly misleading. Namely, not all distribution pairs have a simple decision boundary nor are the probabilities  $\Pr[x | \mathcal{H}_i]$  efficiently computable. In short, likelihood ratio tests are often intractable and thus it makes sense to consider only computationally feasible tests. Random variables  $X$  and  $Y$  are  $(t, \varepsilon)$ -*indistinguishable* if for any  $t$ -time algorithm  $\mathcal{A}$

$$\text{Adv}_{X,Y}^{\text{ind}}(\mathcal{A}) = |\Pr[x \leftarrow X : \mathcal{A}(x) = 1] - \Pr[y \leftarrow Y : \mathcal{A}(y) = 1]| \leq \varepsilon \quad . \quad (3.8)$$

We emphasise that the computational effort needed to create a sample  $x$  is irrelevant, as the running time of  $\mathcal{A}$  depends only on its input and not on the way the input was created. The *computational distance*  $\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1)$  between two hypotheses  $\mathcal{H}_0$  and  $\mathcal{H}_1$  w.r.t. the variable  $x$  is the minimal value<sup>2</sup> of  $\varepsilon$  such that the corresponding distributions are  $(t, \varepsilon)$ -indistinguishable. Observe that there is a correspondence between statistical and computational distance:

$$\text{sd}_x(\mathcal{H}_0, \mathcal{H}_1) = \lim_{t \rightarrow \infty} \text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1) . \quad (3.9)$$

Secondly, we can easily derive a new triangle inequality

$$\text{Adv}_{\mathcal{H}_0, \mathcal{H}_2}^{\text{ind}}(\mathcal{A}) \leq \text{Adv}_{\mathcal{H}_0, \mathcal{H}_1}^{\text{ind}}(\mathcal{A}) + \text{Adv}_{\mathcal{H}_1, \mathcal{H}_2}^{\text{ind}}(\mathcal{A}) , \quad (3.10)$$

from the classical triangle inequality  $|a - c| \leq |a - b| + |b - c|$  by substituting  $a = \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_0]$ ,  $b = \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1]$  and  $c = \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_2]$  into it. As the inequality holds for any algorithm  $\mathcal{A}$  and any time bound  $t$ , we have

$$\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_2) \leq \text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1) + \text{cd}_x^t(\mathcal{H}_1, \mathcal{H}_2) . \quad (3.11)$$

Thirdly, note that if  $\mathcal{H}_0$  and  $\mathcal{H}_1$  determine different distributions, there exists a value  $x_0$  such that  $\Pr[x_0 | \mathcal{H}_0] \neq \Pr[x_0 | \mathcal{H}_1]$ . Consequently, an algorithm  $\mathcal{A}_{x_0}$  that outputs 1 only if  $x = x_0$  and 0 otherwise achieves

$$\text{Adv}_{\mathcal{H}_0, \mathcal{H}_1}^{\text{ind}}(\mathcal{A}_{x_0}) = |\Pr[x_0 | \mathcal{H}_0] - \Pr[x_0 | \mathcal{H}_1]| > 0 . \quad (3.12)$$

Now, if the time bound  $t$  is large enough to compare an input  $x$  and the hard-wired constant  $x_0$ , then the operator  $\text{cd}_x^t(\cdot, \cdot)$  is positively definite

$$\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1) > 0 \iff \mathcal{H}_0 \neq \mathcal{H}_1 . \quad (3.13)$$

As  $\text{cd}_x^t(\cdot, \cdot)$  is clearly symmetric, we have indeed established that  $\text{cd}_x^t(\cdot, \cdot)$  is a distance measure between output distributions for sufficiently large  $t$ .

### 3.2 NEGLIGIBLE EVENTS AND SEMANTIC SECURITY

Computational distance between  $\mathcal{H}_0$  and  $\mathcal{H}_1$  characterises the selectivity of feasible computational tests. If  $\mathcal{A}$  is a  $t$ -time test then analogously to the inequality (3.6) we can prove a lower bound  $\gamma(\mathcal{A}) \geq 1 - \text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1)$ . However, the bound on aggregate error is very non-intuitive. Hence, we pose the question differently and ask whether the best feasible computational test is worth implementing at all. Here, we view cryptology as a tool to make economically sound and scientifically justified decisions. Usually, decisions are made by ignoring events with insignificant probabilities. For example, a probability of an airplane crash is roughly  $2^{-23}$  but most of us are still willing to use air transportation.

More formally, let *negligible* denote a threshold for probabilities such that a change of probability by a negligible amount does not affect economical decisions. Clearly, the threshold depends on the exact setting. Nevertheless, the famous Borel's Law gives some rough estimates: probability  $2^{-20}$  is negligible on the personal scale, probability  $2^{-50}$  is negligible on the terrestrial scale and

<sup>2</sup>Since there are only finite number of  $t$ -time algorithms, the minimum always exists.

probability  $2^{-166}$  is negligible on the cosmic scale [Bor62]. Note that many authors use the term negligible also for characterising the asymptotic growth rate of variables. In such settings, we always use the term *asymptotically negligible* in order to avoid confusion, see Section 2.1 for the precise definition.

Let us now return to simple hypothesis testing and consider the classical scenario where the null hypothesis  $\mathcal{H}_0$  holds with a probability  $\Pr[\mathcal{H}_0] \geq \Pr[\mathcal{H}_1]$ . Then a  $t$ -time test  $\mathcal{A}$  provides a correct answer with probability

$$\begin{aligned} \Pr[\text{success}] &= \Pr[\mathcal{H}_0] \cdot \Pr[\mathcal{A}(x) = 0 | \mathcal{H}_0] + \Pr[\mathcal{H}_1] \cdot \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1] \\ &\leq \Pr[\mathcal{H}_0] + \Pr[\mathcal{H}_1] \cdot (\Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1] - \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_0]) \\ &\leq \Pr[\mathcal{H}_0] + \Pr[\mathcal{H}_1] \cdot \text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1) . \end{aligned} \tag{3.14}$$

Now  $\Pr[\text{success}] = \Pr[\mathcal{H}_0]$  for a trivial test  $\mathcal{A}_* \equiv 0$  and the optimal  $t$ -time test  $\mathcal{A}$  can exceed the trivial success probability only by  $\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1)$ . Consequently, the effort needed to implement  $\mathcal{A}$  is not justified if  $\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1)$  is negligible.

Simple hypothesis testing is often inadequate for many real world scenarios. Usually there are many plausible secret inputs  $s$  instead of two alternatives and a partial disclosure of secrets is also harmful. Thus, given a public input  $f(s)$  it should be infeasible to reliably predict the output  $g(s)$  for any non-constant function  $g$ . Such inability to infer non-trivial information about secret inputs is called *semantic security*. The corresponding security notion was first proposed by Goldwasser and Micali in the context of probabilistic encryption [GM82] and later extensively studied by others [Yao82, MRS88, Lub96, BDJR97].

The concept of semantic security can be modelled in the framework of hypothesis testing. Let  $g(\cdot)$  be a deterministic function. Then any hypothesis about  $g(s)$  naturally splits all plausible values of  $s$  into two sets  $\mathcal{S}_0$  and  $\mathcal{S}_1$  so that the null hypothesis  $\mathcal{H}_0$  holds for all  $s \in \mathcal{S}_0$  and the complementary alternative hypothesis  $\mathcal{H}_1$  for all  $s \in \mathcal{S}_1$ . Strictly speaking, the hypotheses  $\mathcal{H}_0$  and  $\mathcal{H}_1$  alone do not determine the output distribution and we must assume the existence<sup>3</sup> of some unknown probability distributions over  $\mathcal{S}_0$  and  $\mathcal{S}_1$ . Then the simple hypothesis testing scenario still adequately describes the process and the inequality (3.14) holds whenever  $\Pr[\mathcal{H}_0] \geq \Pr[\mathcal{H}_1]$ . Note that we can only upper bound the value of  $\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1)$ , as the distributions on  $\mathcal{S}_0$  and  $\mathcal{S}_1$  are unknown. Let  $\mathcal{H}_{[s=s_i]}$  denote an elementary hypothesis that  $s = s_i$  for  $i \in \{0, 1\}$ . Then

$$\text{cd}_x^t(\mathcal{H}_0, \mathcal{H}_1) \leq \sup_{s_i \in \mathcal{S}_i} \text{cd}_x^t(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_1]}) , \tag{3.15}$$

since we can upper bound  $|\Pr[\mathcal{A}(x) = 1 | \mathcal{H}_1] - \Pr[\mathcal{A}(x) = 1 | \mathcal{H}_0]|$  by

$$\sum_{s_0 \in \mathcal{S}_0} \sum_{s_1 \in \mathcal{S}_1} \Pr[s_0] \cdot \Pr[s_1] \cdot |\Pr[\mathcal{A}(x) = 1 | s_1] - \Pr[\mathcal{A}(x) = 1 | s_0]| \tag{3.16}$$

and upper bound the absolute value by  $\sup_{s_i \in \mathcal{S}_i} \text{cd}_x^t(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_1]})$ . By combining the inequalities (3.14) and (3.15), we obtain the final upper bound

$$\Pr[\text{success}] \leq \Pr[\mathcal{H}_0] + \Pr[\mathcal{H}_1] \cdot \sup_{s_i \in \mathcal{S}_i} \text{cd}_x^t(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_1]}) . \tag{3.17}$$

<sup>3</sup>In our context, such a semi-Bayesian assumption is completely natural although orthodox statisticians would use the notion of universally most powerful tests instead.

The inequality (3.17) holds also for randomised functions  $g(s) = g(s; \omega)$  when the random choices of  $f(s)$  and  $g(s)$  are independent. For proof, we must just consider elementary hypotheses  $\mathcal{H}_{[s=s_i, \omega=\omega_i]}$  instead of  $\mathcal{H}_{[s=s_i]}$ , repeat the derivation and use the independence assumption to verify

$$\text{cd}_x^t(\mathcal{H}_{[s=s_0, \omega=\omega_0]}, \mathcal{H}_{[s=s_1, \omega=\omega_1]}) = \text{cd}_x^t(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_1]}) . \quad (3.18)$$

Finally, consider the scenario where an adversary does not test hypotheses but predicts the output of  $g : \{0, 1\}^* \rightarrow \{0, \dots, n\}$  instead. Let  $\mathcal{H}_i$  denote the hypothesis  $g(s) = i$  and  $\mathcal{S}_i = \{s : g(s) = i\}$ . For simplicity, assume  $\Pr[\mathcal{H}_0] \geq \Pr[\mathcal{H}_i]$  for  $i \in \{1, \dots, n\}$ . Then we can derive

$$\Pr[\text{success}] \leq \Pr[\mathcal{H}_0] + \sum_{k=1}^n \Pr[\mathcal{H}_k] \cdot (\Pr[\mathcal{A}(x) = k | \mathcal{H}_k] - \Pr[\mathcal{A}(x) = k | \mathcal{H}_0]) ,$$

where the terms in the parenthesis are again averages over unknown but existing distributions. Obviously, we can test  $\mathcal{A}(x) = k$  in time  $O(\log n)$  and thus

$$|\Pr[\mathcal{A}(x) = k | \mathcal{H}_k] - \Pr[\mathcal{A}(x) = k | \mathcal{H}_0]| \leq \sup_{s_i \in \mathcal{S}_i} \text{cd}_x^\tau(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_k]}) , \quad (3.19)$$

where  $\tau = t + O(\log n)$ . Therefore, we can conclude

$$\Pr[\text{success}] \leq \Pr[\mathcal{H}_0] + \Pr[\neg \mathcal{H}_0] \cdot \sup_{s_0, s_1} \text{cd}_x^\tau(\mathcal{H}_{[s=s_0]}, \mathcal{H}_{[s=s_1]}) , \quad (3.20)$$

where the supremum is taken over all possible secret inputs. To summarise, we have established that for any function  $g(s)$  with a reasonable output length, the inability to distinguish simple hypotheses implies the inability to predict the output significantly better than choosing the most probable output. Hence, we have established the classical result that indistinguishability implies semantic security. Still, note that the proof technique is strictly non-constructive and the function  $g(\cdot)$  itself can be intractable. The latter is somewhat different from the strictly constructive approach pursued in the article [BDJR97]. We re-examine these issues in Sections 6.6 and 6.7 in the context of subjective security.

### 3.3 INTERACTIVE INFERENCE AND SECURITY GAMES

Thus far we have considered non-interactive inference where an adversary has no control over public outputs. This model is adequate only if the adversary is an external party in the computational process, for example listens to network traffic or breaks into a computational node after the computations have been completed. Alternatively, the adversary can actively participate in the computations and thus influence the sampling procedure itself. Such interactive scenarios are known as games or experiments. Although games have been implicitly around since 1982 when Goldwasser, Micali and Yao first used hybrid arguments [GM82, Yao82], a more formal treatment of games is rather recent. It gradually emerged together with the notion of exact security and reached its maturity around 2004, when several game-playing proof methodologies [BR04, Sho04, Hal05] were explicitly outlined. In some sense, this was an inevitable change forced by the ever-growing complexity of cryptographic proofs and the nit-picking argumentation style demanded by exact security.

Although the basic ideas behind game-playing proofs are rather simple, it is always nice to start with illustrative examples. Various security definitions of public-key cryptosystems are a good starting point, since they have a simple structure and are still complex enough to illustrate all the basic ideas.

**Security of cryptosystems as an example.** A public-key cryptosystem is specified by three efficient randomised algorithms  $\text{Gen}$ ,  $\text{Enc}$ ,  $\text{Dec}$ . A key generation algorithm  $\text{Gen}$  outputs a key pair that consists of a secret key  $\text{sk}$  and a public key  $\text{pk}$ . Two other algorithms  $\text{Enc}_{\text{pk}} : \mathcal{M} \rightarrow \mathcal{C}$  and  $\text{Dec}_{\text{sk}} : \mathcal{C} \rightarrow \mathcal{M} \cup \{\perp\}$  are used to encrypt and decrypt messages. Sets  $\mathcal{M}$  and  $\mathcal{C}$  are known as message and ciphertext spaces, respectively. A cryptosystem is functional if, for all key pairs  $(\text{pk}, \text{sk}) \leftarrow \text{Gen}$  and messages  $m \in \mathcal{M}$ , the equality

$$\text{Dec}_{\text{sk}}(\text{Enc}_{\text{pk}}(m)) = m \quad (3.21)$$

holds with overwhelming probability. Message and ciphertext space can depend on the public key  $\text{pk}$ . Moreover, the decryption algorithm  $\text{Dec}_{\text{sk}}(\cdot)$  can return an error value  $\perp$  if non-ciphertext is used as an input.

Now, there are many security definitions for cryptosystems but all of them are formalised using specific attack scenarios. Goldwasser and Micali were the first to formalise the weakest classical security notion as *indistinguishability under chosen plaintext attack* (IND-CPA security), see [GM84]. In a chosen plaintext attack, an adversary  $\mathcal{A}$  has partial control over the encrypted messages. The corresponding attack scenario is formally captured by security games  $\mathcal{G}_0$  and  $\mathcal{G}_1$  that have oracle access to a stateful *adversarial* algorithm  $\mathcal{A}$ .

$$\begin{array}{l} \mathcal{G}_0^{\mathcal{A}} \\ \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ \text{return } \mathcal{A}(c) \end{array} \right. \end{array} \quad \begin{array}{l} \mathcal{G}_1^{\mathcal{A}} \\ \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ \text{return } \mathcal{A}(c) \end{array} \right. \end{array}$$

Since random variables  $\mathcal{G}_0^{\mathcal{A}}$  and  $\mathcal{G}_1^{\mathcal{A}}$  describe the output of  $\mathcal{A}$  under two different hypotheses, we can generalise the indistinguishability definition. We say that a cryptosystem is  $(t, \varepsilon)$ -IND-CPA secure if for any stateful  $t$ -time adversary  $\mathcal{A}$

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A}) = |\Pr[\mathcal{G}_0^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_1^{\mathcal{A}} = 1]| \leq \varepsilon . \quad (3.22)$$

Sometimes an adversary has limited access to the decryption procedure and thus can conduct chosen ciphertext attacks. Therefore, we can also consider *indistinguishability under chosen ciphertext attacks* (IND-CCA security). There are two possible formalisations: IND-CCA1 security proposed by Naor and Yung [NY90] and IND-CCA2 security proposed by Rackoff and Simon [RS91]. The corresponding indistinguishability games are very similar to the IND-CPA games except  $\mathcal{A}$  has black-box access to oracles  $\mathcal{O}_1$  and  $\mathcal{O}_2$ .

$$\begin{array}{l} \mathcal{G}_0^{\mathcal{A}} \\ \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}_1}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ \text{return } \mathcal{A}^{\mathcal{O}_2}(c) \end{array} \right. \end{array} \quad \begin{array}{l} \mathcal{G}_1^{\mathcal{A}} \\ \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}^{\mathcal{O}_1}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ \text{return } \mathcal{A}^{\mathcal{O}_2}(c) \end{array} \right. \end{array}$$

In the IND-CCA2 games both oracles  $\mathcal{O}_1$  and  $\mathcal{O}_2$  decrypt inputs, i.e.  $\mathcal{O}_i(\hat{c}) = \text{Dec}_{\text{sk}}(\hat{c})$ . To avoid trivial attacks, the game is halted with  $\perp$  if  $\mathcal{A}$  submits the challenge ciphertext  $c$  to  $\mathcal{O}_2$ . In IND-CCA1 game, the oracle  $\mathcal{O}_1$  decrypts inputs and  $\mathcal{O}_2$  does nothing. A cryptosystem is  $(t, \varepsilon)$ -IND-CCA1 or  $(t, \varepsilon)$ -IND-CCA2 secure if for any  $t$ -time adversary  $\mathcal{A}$  the inequality (3.22) holds.

**Basic properties of games.** Formally, a game is a two-party protocol between an honest challenger  $\mathcal{G}$  and a malicious adversary  $\mathcal{A}$  that satisfies some special rules. Most importantly, the challenger and the adversary are executed in turns, so that they are not active at the same time. The game is started by  $\mathcal{G}$  that sends some message to  $\mathcal{A}$  and then stops. Next  $\mathcal{A}$  wakes up reads the message, composes a reply and stops. Then  $\mathcal{G}$  wakes up, reads the message, composes a reply and stops, and so on. The ping-pong with messages continues until either  $\mathcal{A}$  halts with  $\perp$  or  $\mathcal{G}$  halts with  $\text{out} \in \{0, 1, \perp\}$ . The output of the game  $\mathcal{G}^{\mathcal{A}}$  is  $\text{out}$ , unless  $\mathcal{A}$  outputs  $\perp$  then  $\mathcal{G}^{\mathcal{A}} = \perp$ . Note that such a ping-pong execution ensures that  $\mathcal{A}$  cannot measure the computational effort made by the challenger and thus eliminates the possibility of timing-based side-channel attacks.<sup>4</sup>

Now it is straightforward to generalise the notion of computational distance to games. The computational distance  $\text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1)$  between games  $\mathcal{G}_0$  and  $\mathcal{G}_1$  is the minimal value  $\varepsilon$  such that for any  $t$ -time algorithm  $\mathcal{A}$

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A}) = |\Pr[\mathcal{G}_0^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_1^{\mathcal{A}} = 1]| \leq \varepsilon . \quad (3.23)$$

The asterisk in the symbol  $\text{cd}_*^t(\cdot, \cdot)$  emphasises the fact that the resulting interaction pattern may vary depending on the actions of the adversary. Clearly,  $\text{cd}_*^t(\cdot, \cdot)$  is a semi-distance, as it is symmetric and the triangle inequality

$$\text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_2) \leq \text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) + \text{cd}_*^t(\mathcal{G}_1, \mathcal{G}_2) . \quad (3.24)$$

follows from the basic triangle inequality  $|a - c| \leq |a - b| + |c - b|$  similar to the inequality (3.11). Again, we can define statistical distance as a limit

$$\text{sd}_*(\mathcal{G}_0, \mathcal{G}_1) = \lim_{t \rightarrow \infty} \text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) . \quad (3.25)$$

We say that games  $\mathcal{G}_0$  and  $\mathcal{G}_1$  are equivalent if  $\text{sd}_*(\mathcal{G}_0, \mathcal{G}_1) = 0$ , since any adversary  $\mathcal{A}$  has an equal probability to win both games  $\mathcal{G}_0$  and  $\mathcal{G}_1$ . Again, we can set restrictions for the games so that for large enough  $t$ , the double implication  $\text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) > 0 \Leftrightarrow \mathcal{G}_0 \not\equiv \mathcal{G}_1$  always holds and  $\text{cd}_*^t(\cdot, \cdot)$  becomes a distance measure. Hence, the name computational distance is still justified.

**Different description styles.** Although a game is just a two-party protocol between an honest challenger  $\mathcal{G}$  and a malicious adversary  $\mathcal{A}$ , different cryptographic formalisms use different ways to describe games, see Fig. 3.2.

One viable option is to fully specify actions of the challenger together with a rigid messaging schedule for the protocol as we specified for IND-CPA games. Such a description style is the oldest and most widespread in cryptographic literature. More formally, a *challenger-centric* description specifies a game as an oracle algorithm  $\mathcal{G}$  that internally calls out a stateful subroutine  $\mathcal{A}$  until  $\mathcal{G}$  reaches some output or is halted due to abnormal behaviour of  $\mathcal{A}$ . Sometimes,

<sup>4</sup>Here, we just eliminate accidental exposure of timing artefacts, since we can re-introduce timing information in a controlled way by modifying the description of the game.

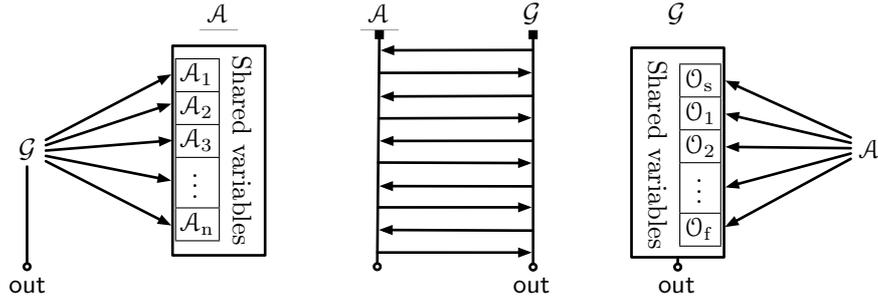


Figure 3.2: Duality between challenger- and adversary-centric game descriptions. Both formalisations describe the interactive process shown in the middle.

the adversary  $\mathcal{A}$  is even represented as a collection of different stateless algorithms  $\mathcal{A}_1, \dots, \mathcal{A}_n$  that communicate by using shared variables.

The main virtue of the challenger-centric description is simplicity. For games with a deterministic scheduling, it is the most natural approach. However, for non-deterministic scheduling, the approach quickly becomes cumbersome and tedious. A reader can easily verify this by formalising the description of IND-CCA2 games without explicit use of decryption oracles.

Another compelling alternative is an *adversary-centric* description of games. Compared to the challenger-centric description, the tables are turned—the algorithm  $\mathcal{A}$  internally calls out a subroutine  $\mathcal{G}$ , until  $\mathcal{G}$  provides the desired output. Formally, the challenger code is split into smaller threads  $\mathcal{O}_s, \mathcal{O}_1, \dots, \mathcal{O}_n, \mathcal{O}_f$  that are scheduled according to the adversary’s requests. The game is started by an oracle  $\mathcal{O}_s$  that initialises all necessary variables and provides input to the adversary  $\mathcal{A}$ . Next,  $\mathcal{A}$  can interact with oracles  $\mathcal{O}_1, \dots, \mathcal{O}_n$  by submitting queries and receiving replies. To end the game,  $\mathcal{A}$  calls out the finalisation procedure  $\mathcal{O}_f$  that computes the output of the game  $\text{out} \in \{0, 1, \perp\}$ . Note that oracles  $\mathcal{O}_s, \mathcal{O}_1, \dots, \mathcal{O}_n, \mathcal{O}_f$  as threads of the same program can communicate through shared variables and thus force constraints on the adversarial behaviour.

As an example, consider the adversary-centric description of the IND-CPA games. The corresponding initialisation oracle  $\mathcal{O}_s$  runs the key generation algorithm  $\text{Gen}$  and stores the corresponding key pair  $(\text{sk}, \text{pk})$ . The oracle  $\mathcal{O}_s$  always replies  $\text{pk}$  to all queries that  $\mathcal{A}$  makes. The second oracle  $\mathcal{O}_1$  can be queried only once. Given a message pair  $(m_0, m_1)$ , the oracle  $\mathcal{O}_1$  replies  $\text{Enc}_{\text{pk}}(m_0)$  or  $\text{Enc}_{\text{pk}}(m_1)$  depending on whether we are in the game  $\mathcal{G}_0$  or in the game  $\mathcal{G}_1$ . The finalisation oracle  $\mathcal{O}_f$  always outputs the first message sent by  $\mathcal{A}$ .

We emphasise that both description styles have their advantages and drawbacks and it is only a matter of taste which syntactic sugar we choose. In short, the challenger-centric approach forces constraints explicitly through message scheduling, whereas the adversary-centric approach forces constraints implicitly through cooperative behaviour of oracles. Generally, the adversary-centric approach is better in the context of complex interactions, since it is well suited for non-deterministic scheduling. However, a suitable compromise between the two approaches can often be even more descriptive and concise.

**Indistinguishability and semantic security.** A game corresponds to an interactive hypothesis testing scenario if the challenger  $\mathcal{G}$  always outputs the last reply from the adversary  $\mathcal{A}$ . Note that the argumentation given in Section 3.2

also holds for interactive hypothesis testing. More precisely, let  $\{\mathcal{G}_s\}_{s \in \mathcal{S}}$  be a set of games such that any pair of them is computationally indistinguishable. In that case, the corresponding proofs in Section 3.2 still hold and consequently bounded adversaries cannot restore even partial information about  $s$ . Hence, confidentiality of secret inputs can be stated in terms of indistinguishability.

Naturally, the exact formulation is not as simple as that. Protocols usually leak some information about secret inputs and thus the pairwise indistinguishability of games is not always achievable. Moreover, interactive hypothesis testing forms only a small subclass of games, i.e., games can specify more complex properties, such as consistency and verifiability of protocol outputs. We discuss these issues further in Chapters 5 and 7.

## 4 CRYPTOGRAPHIC PROOF TECHNIQUES

Games are powerful tools for analysing interactive computations, as they are flexible enough to capture any attack pattern needed to define a specific design goal, such as authenticity, confidentiality or fairness. Moreover, the strength of a cryptographic construction is often quantified as an advantage

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}) = \Pr [\mathcal{G}^{\mathcal{A}} = 1] \quad (4.1)$$

against a specific game  $\mathcal{G}$ . Therefore, we must often upper bound advantages in order to prove the security of a primitive or a protocol. To establish these bounds, we can use either direct or conditional proofs. Direct proofs use Peano axiomatisation of arithmetic and nothing more, whereas conditional proofs rely on additional assumptions, such as the hardness of factoring or  $\mathbf{P} \neq \mathbf{NP}$ .

Note that direct proofs are somewhat trivial in the settings where the running times of all participants are bounded. Since there is only a finite number of different adversaries and a finite number of relevant random bits, maximising the advantage  $\text{Adv}_{\mathcal{G}}(\mathcal{A})$  is a discrete optimisation task over a finite set. Hence, for most cryptographic problems there exists either an exhaustive proof or an explicit counterexample. However, the verification time for such proofs is gargantuan, compared to the time bounds established in them. Hence, direct proofs are useful only if they are compact enough to be efficiently verifiable.

Unfortunately, no compact direct proofs have been discovered so far in cryptography. One of the few non-trivial results is a construction of a permutation that can be inverted approximately 2 times slower than computed [Hil92] and any significant advance in this field is believed to be hard [Mas96].

As a way out, we take the existence of certain cryptographic primitives for granted and build everything else on top of these basic primitives. As a result, we can use indirect conditional proofs to establish security. More precisely, it is sufficient to prove that a construction can be insecure only if some basic primitive fails to meet the specification. Moreover, the approach separates abstract design goals from practical implementation details. Still, we cannot completely ignore complexity theoretical details, or otherwise we end up with basic primitives that cannot be implemented at all. Consequently, the choice of basic primitives should always reflect our *beliefs* in various complexity theoretical conjectures. Also, note that the choice of basic primitives determines the abstraction level. For example, we can use basic primitives to model high-level properties, such as one-way functions or IND-CPA secure cryptosystems, or more specific computational assumptions, such as intractability of factoring RSA composites.

In this chapter, we show how to decompose conditional proofs into small elementary steps that are easy to apply and verify. The verification procedure itself can be completely formalised. In fact, it is possible to construct a hypothetical proof system CRYPROOF that automatically verifies a proof and computes the corresponding security guarantees, see the discussion in Section 4.5.

## 4.1 REDUCTIONS AS REWRITING RULES

Although the use of abstract security properties simplifies proofs and makes them more universal, it also brings along some drawbacks. In particular, such a choice restricts the structure of possible proofs. Let us consider the IND-CPA cryptosystem as an illustrative example. Recall that a public-key cryptosystem is specified by efficient algorithms  $\text{Gen}, \text{Enc}, \text{Dec}$  and a functional requirement that  $\text{Dec}_{\text{sk}}(\text{Enc}_{\text{pk}}(m)) \equiv m$  for every key pair  $(\text{sk}, \text{pk}) \leftarrow \text{Gen}$ . There are no restrictions as to how a cryptosystem must operate or what its internal structure is. Hence, a new construction built on top of the  $(t, \varepsilon)$ -IND-CPA cryptosystem cannot use any other operations than  $\text{Gen}, \text{Enc}, \text{Dec}$  to manipulate keys, messages and ciphertexts. Of course, such restrictions do not extend to potential adversaries. As the construction assumes nothing about the implementation details, the provable security guarantees must hold for all implementations of  $(t, \varepsilon)$ -IND-CPA secure cryptosystems regardless of how bizarre they might seem. Thus, the only way to prove security is to show that the insecurity of the construction implies an explicit contradiction with the IND-CPA security premise.

More generally, the abstraction of security properties always leads to indirect proofs with identical structure where one must prove  $\text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_1) \leq \varepsilon_1$  from premises of type  $\text{cd}_*^{t_0}(\mathcal{Q}_0, \mathcal{Q}_1) \leq \varepsilon_0$ . The corresponding indirect proof shows that an assumption  $\text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_1) > \varepsilon_1$  implies a contradiction  $\text{cd}_*^{t_0}(\mathcal{Q}_0, \mathcal{Q}_1) > \varepsilon_0$ . The heart of the proof is a code transformation rule (*reduction*), which converts any  $t_1$ -time adversary  $\mathcal{A}$  to a  $t_0$ -time adversary  $\mathcal{B}$  such that

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A}) > \varepsilon_1 \quad \implies \quad \text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathcal{B}) > \varepsilon_0 . \quad (4.2)$$

Most of the reductions are *black-box reductions*, where  $\mathcal{B}$  internally runs  $\mathcal{A}$  in the black-box manner to play the games  $\mathcal{Q}_0$  and  $\mathcal{Q}_1$ . That is,  $\mathcal{B}$  analyses only the messages that  $\mathcal{A}$  writes on the communication and input-output tapes. However,  $\mathcal{B}$  controls all external factors influencing  $\mathcal{A}$ : inputs, random coins, received messages and used timers. In particular,  $\mathcal{B}$  can always restart  $\mathcal{A}$  with different inputs, replies and random coins or just rewind  $\mathcal{A}$  back to any other suitable state. In *white-box reductions*, the dependence between  $\mathcal{A}$  and  $\mathcal{B}$  can be arbitrary, for example  $\mathcal{B}$  can actively alter the internal variables of  $\mathcal{A}$ .

If game pairs  $(\mathcal{G}_0, \mathcal{G}_1)$  and  $(\mathcal{Q}_0, \mathcal{Q}_1)$  are similar enough, it is often advantageous to construct a reduction where  $\mathcal{B}$  simulates the game  $\mathcal{G}_i$  to  $\mathcal{A}$ . Fig. 4.1 depicts an illustrative example where games  $\mathcal{G}_0$  and  $\mathcal{G}_1$  differ only by a single line  $c \leftarrow \text{Enc}_{\text{pk}}(m_0)$  versus  $c \leftarrow \text{Enc}_{\text{pk}}(m_1)$ . More formally, in the game  $\mathcal{G}_i$  the challenger generates  $(\text{sk}, \text{pk}) \leftarrow \text{Gen}$ , interacts with  $\mathcal{A}$  and somehow obtains a valid message pair  $m_0, m_1 \in \mathcal{M}$ . Next, the challenger computes an encryption  $c \leftarrow \text{Enc}_{\text{pk}}(m_i)$  and then continues with the game  $\mathcal{G}_i$ . Now consider an adversary  $\mathcal{B}$  against IND-CPA games that first obtains  $\text{pk}$  from the challenger and then simulates the game  $\mathcal{G}_i$  to  $\mathcal{A}$  until the line  $c \leftarrow \text{Enc}_{\text{pk}}(m_i)$  is reached. Next  $\mathcal{B}$  outputs  $(m_0, m_1)$  and uses the reply as  $c$  and finally outputs  $\mathcal{G}_i^{\mathcal{A}}$ .

Note that the simulation of  $\mathcal{G}_i$  is possible only if  $\text{sk}$  is not used in the game. Similarly, the line  $c \leftarrow \text{Enc}_{\text{pk}}(m_i)$  must be reachable only once, or otherwise  $\mathcal{B}$  violates the rules of IND-CPA games  $\mathcal{Q}_i$ . If these two preconditions are satisfied,  $\mathcal{B}$  simulates the game  $\mathcal{G}_i$  perfectly and thus  $\Pr[\mathcal{G}_i^{\mathcal{A}} = 1] = \Pr[\mathcal{Q}_i^{\mathcal{B}} = 1]$ . Consequently, we have proven that  $\text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) \leq \varepsilon$  if the cryptosystem is  $(t + \tau + O(1), \varepsilon)$ -IND-CPA secure, where  $\tau$  is the running time of  $\mathcal{G}_i$ . More precisely,

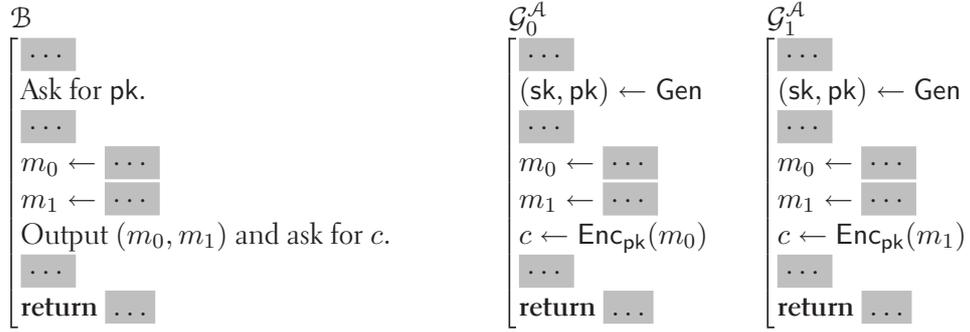


Figure 4.1: A schematic description of an IND-CPA reduction. By construction, the adversary  $\mathcal{B} = \mathcal{B}^{\mathcal{A}}$  on the left simulates perfectly a game  $\mathcal{G}_i^{\mathcal{A}}$  on the right.

the claim only holds for RAM machines, since the simulation of  $\mathcal{G}_i^{\mathcal{A}}$  can take more time on a Turing machine due to tape rewinding.

Note that the reasoning above holds for any game pair  $(\mathcal{G}_0, \mathcal{G}_1)$  that satisfies the preconditions. Moreover, such semi-transparent reasoning is common in cryptography. In fact, any valid reduction proof implicitly specifies a code transformation  $\mathcal{T}$  that preserves computational closeness under certain preconditions  $\mathcal{P}$ . The closeness guarantee  $\mathcal{S}$  is often in the form: if the condition  $\mathcal{P}$  holds for  $\mathcal{G}_0$  and  $\mathcal{G}_1 = \mathcal{T}(\mathcal{G}_0)$  then  $\text{cd}_x^t(\mathcal{G}_0, \mathcal{G}_1) \leq \varepsilon$ . For such reductions, we use shorthands  $\mathcal{G}_0 \xrightarrow[\varepsilon]{\mathcal{T}} \mathcal{G}_1$  to emphasise the bound on the computational distance.

**Definition 1.** A reduction schema is a triple  $(\mathcal{P}, \mathcal{T}, \mathcal{S})$ , where  $\mathcal{P}$  is a precondition,  $\mathcal{T}$  is a code transformation and  $\mathcal{S}$  is the corresponding security guarantee.

These reduction schemata significantly simplify the derivation of security proofs. It is much easier to test whether a certain precondition holds than to construct the same reduction over and over again. However, we should note that reduction schemata are not usually rigorously formalised and applied, rather they are used informally to construct complex reductions as proofs.

Generally, each basic primitive introduces at least one reduction schema, but there can be more than one schema for each of them. Since reductions schemata can have different preconditions and security guarantees, the major task in cryptography is to identify triples  $(\mathcal{P}, \mathcal{T}, \mathcal{S})$  that are optimal, i.e.,  $\mathcal{P}$  cannot be weakened or  $\mathcal{T}$  generalised without changing  $\mathcal{S}$ . Note that compiling a security proof, given a proper set of optimal reduction schemata, is just a plain engineering task. In other words, such an approach can be viewed as a further modularisation of cryptographic proofs into trivial and non-trivial portions.

## 4.2 REDUCTIONS AND TIME-SUCCESS PROFILES

Any attack against a cryptographic construction or a protocol is a trade-off between desired goals and available resources. It is possible to break any construction by applying a sufficient amount of computational resources, unless it is information-theoretically secure. At the same time, additional computational resources increase the cost of the attack and thus a rationally behaving attacker must find a proper balance between estimated losses and potential gains.

The time-success profile is the most obvious way to describe trade-offs between the losses and gains. For many security objectives, we can construct a game  $\mathcal{G}$  such that the advantage  $\text{Adv}_{\mathcal{G}}(\mathcal{A}) = \Pr[\mathcal{G}^{\mathcal{A}} = 1]$  quantifies the success probability of an adversary  $\mathcal{A}$ . Now a *time-success profile* is a function

$$\varepsilon(t) = \max_{\mathcal{A} \in \mathfrak{A}(t)} \text{Adv}_{\mathcal{G}}(\mathcal{A}) , \quad (4.3)$$

where  $\mathfrak{A}(t)$  consists of all  $t$ -time adversaries. Note that the time-success profile is well-defined for every concrete construction or primitive. In fact, a time-success profile is even computable by exhaustive inspection of all possibilities. However, such a process is clearly intractable for all practical constructions and thus we can only approximate time-success profiles. One possibility is to view a cryptographic reduction as a rule that combines time-success profiles of basic primitives into a time-success profile of a new construction. More precisely, we get an upper bound on the time-success profile in terms of basic primitives.

Unfortunately, even the approximate shapes of time-success profiles are unknown for all cryptographic primitives used in practice. Only some indicative results have been established for uninstantiable generic models. For example, the time-success profile of the discrete logarithm and Diffie-Hellman problem is known to be  $\Theta(t^2)$  in generic groups [Nec94, Sho97, SJ00]. Analogous results are known for the RSA root finding problem in generic rings [DK02, LR06].

Time-success ratio is a more conservative alternative for approximating time-success profiles. The term itself was first proposed in the context of asymptotic security [HL92, Lub96], but it has a natural interpretation also for a fixed primitive in the exact security setting. Namely, one can estimate the minimal *time-success ratio*  $\alpha$  for a primitive and thus fix a linear upper bound:

$$\alpha = \min_t \frac{t}{\varepsilon(t)} \quad \iff \quad \forall t : \varepsilon(t) \leq \frac{t}{\alpha} . \quad (4.4)$$

If the time-success profile has a globally convex-cup shape, then  $\alpha$  is the minimum time needed to completely break the primitive, see Fig. 4.2. For many low level primitives, the adversary can compute the output  $\mathcal{G}^{\mathcal{A}}$  by himself, e.g., verify that he or she has factored a large number or has found a discrete logarithm. For such primitives, the time-success profile becomes approximately convex-cup if the adversary can restart<sup>1</sup> the security game at any point. Then an estimate on the minimal breaking time is a theoretically sound estimate of  $\alpha$ .

A time-success profile captures the essential information about the cryptographic construction. In particular, one can assess how secure the construction is in real world scenarios. For a moment, consider the security against rational adversaries that always choose the most beneficial attack. To analyse such behaviour, we must rescale the time and success axes so that they represent expected losses and gains, see Fig. 4.2. Obviously, the cost of an attack depends on the computational complexity. Let  $x(t)$  be the correspondence between running time and computing costs. Similarly, there is a connection between success probability and the potential utility  $y(\varepsilon)$  in dollars. Note that besides direct objective costs, an adversary may have indirect subjective costs, such as a threat

<sup>1</sup>If the problem is randomly self-reducible, the adversary can restart the game internally. Therefore, the profile is approximately convex-cup for the discrete logarithm problem.

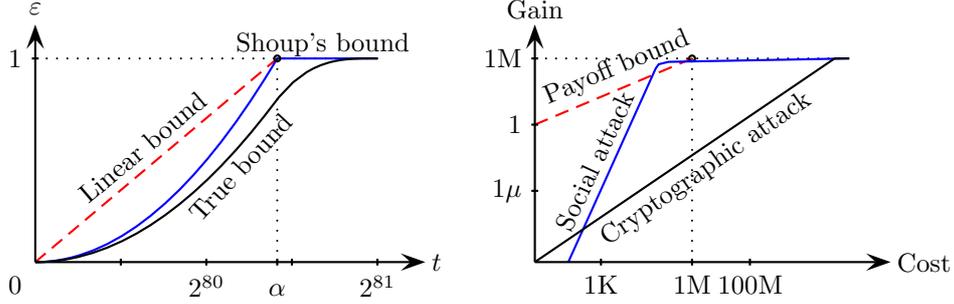


Figure 4.2: A hypothetical time-success profile for a discrete logarithm problem and the corresponding cost-gain profile in the case of 1 million dollar award. The left graph is given in the linear and the right in the logarithmic scale.

to ruin his or her reputation by attacking. Analogously, the correspondence between success probability and potential gains may be nonlinear.

Evidently, a construction is secure against rational adversaries if the costs are always greater than the potential gains. Since an adversary may also conduct physical and social engineering attacks, the whole system may still be insecure, even if the cryptographic construction is secure, see Fig. 4.2. Ideally, the cryptographic strength should be comparable to other weaknesses, as the use of overly complex cryptographic primitives leads to unjustified maintenance costs.

Not all entities behave rationally. Some of them may want to cause maximal achievable damage even at any cost. To analyse such scenarios, we must turn the tables and consider the situation from the perspective of legitimate users. In particular, we must use different rescaling such that the  $x(t)$  axis still represents the potential costs of an attack, whereas  $y(\varepsilon)$  counts potential losses of legitimate users, including the resources needed to run the system. As a result, we can estimate the losses in terms of available resources and thus compare different protection mechanisms with different profiles. In particular, it may turn out that the most cost-efficient solution is to have no protection at all.

Since time-success profiles are so important, it is crucial to understand how cryptographic reductions change these profiles. In the simplest case, one can prove that a  $(t_1, \varepsilon_1)$ -attack against a construction can be converted to a  $(t_0, \varepsilon_0)$ -attack against the basic primitive. The corresponding *tightness factor*

$$\beta(\varepsilon_1, t_1) = \frac{\alpha_1}{\alpha_0} = \frac{t_1 \varepsilon_0}{t_0 \varepsilon_1} \quad (4.5)$$

characterises the inevitable security loss, i.e., how much does the bound on the total breaking time decrease. Again, the tightness factors have been used in the context of asymptotic security [HL92, Lub96], but the corresponding classification [Lub96, p. 21–34] of reductions is too coarse for our purposes.

For more precise classification, consider the correspondence between time-success profiles. Fix a linear target bound  $\varepsilon_1 = \Theta(t_1)$ . In such case, a reduction is *linear* if it leads to a linear bound  $\varepsilon_0 = \Theta(t_0)$ . Most cryptographic reductions are linear but complex reductions can lead to convex-cup bounds  $\varepsilon_0 = \Theta(t_0^k)$ . The constant  $k \geq 1$  determines the order of the reduction, for example, a reduction is *quadratic* if  $\varepsilon_0 = \Theta(t_0^2)$ . Since we normally start with linear bounds  $\varepsilon_0 = \Theta(t_0)$  for basic primitives, we get convex-cap security guarantees  $\varepsilon_1 = \Theta(\sqrt[k]{t})$  for the derived construction or primitive. Such a shape is

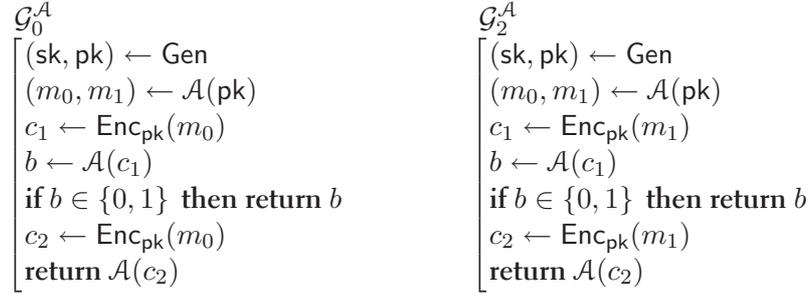


Figure 4.3: Two-stage attack against a public-key cryptosystem.

quite unsuitable from game theoretical viewpoint, as it increases the potential payoff in a low costs range and also increases the bound on  $\alpha_1$ .

Sometimes, it is possible to provide several reductions for the same problem. Often, these reductions have incomparable structure and security bounds. Then the comparison of tightness factors  $\beta(\varepsilon_1, t_1)$  provides some insight. Assume that basic primitives have linear time-success bounds. Then a reduction with the largest tightness ratio allows us to choose a primitive with the smallest  $\alpha_0$  and thus we gain efficiency without changing the security level.

Finally, the tightness ratio characterises the robustness of the bounds. As all current estimates on the security of cryptographic primitives are heuristic, they are likely to change in the future. The ratio  $\beta$  characterises how big the corresponding changes in derived security guarantees are.

### 4.3 SURPRISING PROPERTIES OF CONDITIONAL PROBABILITIES

Before we investigate standard proof techniques, it is instructive to consider common dangers first. In particular, note that conditional probabilities often behave counter-intuitively and the corresponding confusion can cause subtle argumentation flaws in security proofs. Let us start with a simple example and analyse games  $\mathcal{G}_0$  and  $\mathcal{G}_2$  in Fig. 4.3 under the assumption that a cryptosystem is  $(t, \varepsilon)$ -IND-CPA secure. Note that the adversary  $\mathcal{A}$  must distinguish between ciphertexts of messages  $m_0$  and  $m_1$  but differently from IND-CPA games, the adversary  $\mathcal{A}$  can choose not to answer. Namely, if  $b \in \{0, 1\}$  then the challenger accepts  $b$  as a final guess, otherwise a new ciphertext  $c_2$  is generated and the adversary  $\mathcal{A}$  gets a second chance. In other words, if  $b \in \{0, 1\}$ , then  $\mathcal{A}$  plays the IND-CPA game with the challenge  $c_1$  and otherwise  $\mathcal{A}$  plays the IND-CPA game with challenge  $c_2$ . Hence, it would be natural to assume that the following inequalities should hold for any  $t$ -time adversary  $\mathcal{A}$ :

$$\text{Adv}_1(\mathcal{A}) = |\Pr[\mathcal{G}_0^{\mathcal{A}} = 1 | b \in \{0, 1\}] - \Pr[\mathcal{G}_2^{\mathcal{A}} = 1 | b \in \{0, 1\}]| \leq \varepsilon, \quad (4.6)$$

$$\text{Adv}_2(\mathcal{A}) = |\Pr[\mathcal{G}_0^{\mathcal{A}} = 1 | b \notin \{0, 1\}] - \Pr[\mathcal{G}_2^{\mathcal{A}} = 1 | b \notin \{0, 1\}]| \leq \varepsilon. \quad (4.7)$$

However, neither of these inequalities (4.6) and (4.7) holds, as efficient adversaries  $\mathcal{A}_1$  and  $\mathcal{A}_2$  depicted in Fig. 4.4 achieve  $\text{Adv}_1(\mathcal{A}_1) = \text{Adv}_2(\mathcal{A}_2) = 1$ . For the proof, note that if messages differ  $m_0 \neq m_1$ , then a collision  $\text{Enc}_{\text{pk}}(m_j) = \text{Enc}_{\text{pk}}(m_i)$  implies  $m_i = m_j$ . Therefore,  $\mathcal{A}_1$  outputs a correct answer if  $b \in \{0, 1\}$

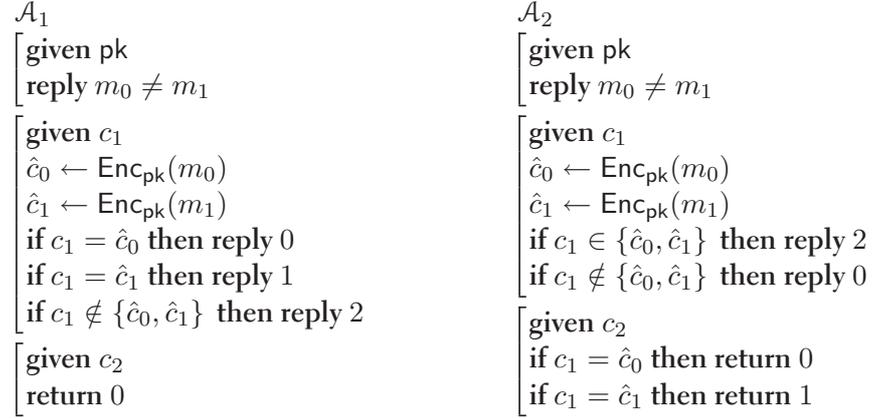


Figure 4.4: Adversaries  $\mathcal{A}_1$  and  $\mathcal{A}_2$  which achieve  $\text{Adv}_1(\mathcal{A}_1) = \text{Adv}_2(\mathcal{A}_2) = 1$ .

and  $\mathcal{A}_2$  outputs a correct answer if  $b \notin \{0, 1\}$ . Hence, we have proven that

$$\Pr[\mathcal{G}_0^{\mathcal{A}_1} = 1 | b \in \{0, 1\}] = 0, \quad \Pr[\mathcal{G}_2^{\mathcal{A}_1} = 1 | b \in \{0, 1\}] = 1, \quad (4.8)$$

$$\Pr[\mathcal{G}_0^{\mathcal{A}_2} = 1 | b \notin \{0, 1\}] = 0, \quad \Pr[\mathcal{G}_2^{\mathcal{A}_2} = 1 | b \notin \{0, 1\}] = 1. \quad (4.9)$$

Note that the drastic discrepancy between superficial estimates (4.6) and (4.7), and true bounds (4.8) and (4.9) is caused by the unintentional mismatch of prior probabilities in the reasoning. Namely, the inequality (4.6) follows from IND-CPA security only if  $\Pr[b \in \{0, 1\}] = 1$ , but the actual probability  $\Pr[b \in \{0, 1\}]$  can be arbitrarily small. Flaws in the derivation of (4.7) are analogous.

A similar flaw, but in a much more subtle form appears in the classical proofs of the PRP/PRF Switching Lemma (Theorem 5.1 in [IR88]). The error itself was discovered alarmingly late, in 2004, by Kohno, see the discussion in [BR04]. The theorem estimates a statistical distance between two games  $\mathcal{G}_0$  and  $\mathcal{G}_1$ . In the game  $\mathcal{G}_0$  the challenger chooses a function  $f$  uniformly from the set of all functions  $f : \mathcal{X} \rightarrow \mathcal{X}$ , whereas  $f$  is a uniformly chosen permutation on  $\mathcal{X}$  in the game  $\mathcal{G}_1$ . The task of an adversary  $\mathcal{A}$  is to distinguish these games by adaptively querying at most  $q$  values of  $f(x_i)$ . Note that any deterministic strategy  $\mathcal{A}$  can be formalised as a tree, where the nodes represent queries  $f(x_i)$  and the edges correspond to the responses as illustrated in Fig. 4.5. In the game  $\mathcal{G}_0$  all paths are possible, whereas no path that leads to a collision  $f(x_i) = f(x_j)$  is possible in  $\mathcal{G}_1$ . In both games, the probability to reach a plausible node depends only on the number of queries. Now if  $\mathcal{A}$  always makes *exactly*  $q$  distinct queries, then

$$\Pr[\mathcal{G}_1^{\mathcal{A}} = 1] = \Pr[\mathcal{G}_0^{\mathcal{A}} = 1 | \neg \text{Collision}] \quad (4.10)$$

as nodes are sampled with uniform probability. However, if  $\mathcal{A}$  decides to stop earlier in some paths, then conditional probabilities  $\Pr[\text{node} | \neg \text{Collision}]$  start to differ between the games. For example, the conditional probabilities from left to right on the decision border in Fig. 4.5 are  $(\frac{0}{7}, \frac{1}{7}, \frac{1}{7}, \frac{0}{7}, \frac{1}{7}, \frac{1}{7}, \frac{3}{7})$  in the game  $\mathcal{G}_0$  and  $(\frac{0}{6}, \frac{1}{6}, \frac{1}{6}, \frac{0}{6}, \frac{1}{6}, \frac{1}{6}, \frac{2}{6})$  in the game  $\mathcal{G}_1$ . Therefore, the equation (4.10) is not guaranteed to hold when  $\mathcal{A}$  halts prematurely in some paths. Consequently, in the strict sense, the proofs given in [IR88, BKR94] are correct as they consider adversaries that make *exactly*  $q$  distinct queries. If we consider adversaries that

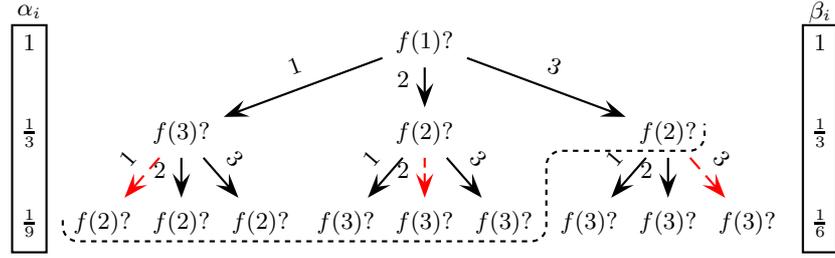


Figure 4.5: A querying strategy for  $\mathcal{X} = \{1, 2, 3\}$  with premature stopping. The dashed line denotes the decision border and  $\alpha_i, \beta_i$  denote the probabilities of plausible queries  $f(x_1), \dots, f(x_i)$  in the games  $\mathcal{G}_0$  and  $\mathcal{G}_1$ , respectively.

can make up to  $q$  queries, then these proofs are indeed incorrect [BR04]. Nevertheless, the final claim is still correct, as adversaries that make up to  $q$  queries cannot outperform adversaries that make exactly  $q$  queries.

To summarise, reasoning about conditional probabilities is counterintuitive and non-robust against microscopic changes in the security claims.

#### 4.4 FROM GAME CHAINS TO PROOF TREES

Let us return to the first example posed in the previous section. Observe that the description of  $\mathcal{G}_0$  and  $\mathcal{G}_2$  differ in two places, namely, ciphertexts  $c_1$  and  $c_2$  are computed differently. As the secret key  $\text{sk}$  is not used in the games, we can apply the IND-CPA reduction schema twice to  $\mathcal{G}_0$  and obtain a game chain

$$\begin{array}{ccc}
 \mathcal{G}_0^A & \mathcal{G}_1^A & \mathcal{G}_2^A \\
 \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c_1 \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ b \leftarrow \mathcal{A}(c_1) \\ \text{if } b \in \{0, 1\} \text{ then } \dots \\ c_2 \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ \text{return } \mathcal{A}(c_2) \end{array} \right. & \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c_1 \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ b \leftarrow \mathcal{A}(c_1) \\ \text{if } b \in \{0, 1\} \text{ then } \dots \\ c_2 \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ \text{return } \mathcal{A}(c_2) \end{array} \right. & \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen} \\ (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c_1 \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ b \leftarrow \mathcal{A}(c_1) \\ \text{if } b \in \{0, 1\} \text{ then } \dots \\ c_2 \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ \text{return } \mathcal{A}(c_2) \end{array} \right.
 \end{array}$$

where the grey boxes emphasise changes. To represent reductions, we can write

$$\mathcal{G}_0 \xrightarrow[\varepsilon]{\text{IND-CPA}} \mathcal{G}_1 \xrightarrow[\varepsilon]{\text{IND-CPA}} \mathcal{G}_2 . \quad (4.11)$$

Now if the cryptosystem is  $(t, \varepsilon)$ -IND-CPA secure, there exists a time bound  $t_1 = t - O(1)$  such that  $\text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_1) \leq \varepsilon$  and  $\text{cd}_*^{t_1}(\mathcal{G}_1, \mathcal{G}_2) \leq \varepsilon$ . Hence, the triangle inequality yields

$$\text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_2) \leq \text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_1) + \text{cd}_*^{t_1}(\mathcal{G}_1, \mathcal{G}_2) \leq 2\varepsilon . \quad (4.12)$$

More generally, we can use game chains as a systematic way to estimate computational distances. Moreover, note that for any  $t$ -time algorithm  $\mathcal{A}$

$$\text{Adv}_{\mathcal{G}_0}(\mathcal{A}) = \Pr[\mathcal{G}_0^A = 1] \leq \text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_n) + \text{Adv}_{\mathcal{G}_n}(\mathcal{A}) \quad (4.13)$$

$$\begin{array}{l}
\mathcal{G}_0^{\mathcal{A}} \\
\left[ \begin{array}{l}
(\text{sk}_0, \text{pk}_0) \leftarrow \text{Gen} \\
(\text{sk}_1, \text{pk}_1) \leftarrow \text{Gen} \\
x \leftarrow \{0, 1\} \\
q \leftarrow \mathcal{A}(\text{pk}_0, \text{pk}_1) \\
\text{if } q \notin \{0, 1\} \text{ then return } \perp \\
c_0 \leftarrow \text{Enc}_{\text{pk}_0}(x) \\
c_1 \leftarrow \text{Enc}_{\text{pk}_1}(x) \\
\text{return } \mathcal{A}(\text{sk}_q, c_0, c_1)
\end{array} \right.
\end{array}
\qquad
\begin{array}{l}
\mathcal{G}_1^{\mathcal{A}} \\
\left[ \begin{array}{l}
(\text{sk}_0, \text{pk}_0) \leftarrow \text{Gen} \\
(\text{sk}_1, \text{pk}_1) \leftarrow \text{Gen} \\
x \leftarrow \{0, 1\} \\
q \leftarrow \mathcal{A}(\text{pk}_0, \text{pk}_1) \\
\text{if } q \notin \{0, 1\} \text{ then return } \perp \\
c_0 \leftarrow \text{Enc}_{\text{pk}_0}(x) \\
c_1 \leftarrow \text{Enc}_{\text{pk}_1}(1 - x) \\
\text{return } \mathcal{A}(\text{sk}_q, c_0, c_1)
\end{array} \right.
\end{array}$$

Figure 4.6: A game pair derived from the Bellare-Micali protocol.

and thus we can use game chains to also upper bound advantages against individual games. Often, such proofs are compressed further by combining the elementary reductions into a single aggregate construction. The latter is known as a constructive hybrid argument, see Section 6.7. However, this is just a trade-off between compactness and clarity: complex reductions are much harder to comprehend and verify than the corresponding elementary reductions.

It is important to note that not all security proofs can be represented as game chains. See Fig. 4.6 contains for a specific counterexample. We remark that the game pair is not arbitrary—slightly more sophisticated game pairs appear in the security proof of the Bellare-Micali oblivious transfer protocol [BM89].

As the adversary  $\mathcal{A}$  can decrypt only a single value  $c_i$ , he or she cannot learn whether  $c_1$  and  $c_2$  are ciphertexts of the same message or not. However, we cannot use the IND-CPA transformation, as we do not know in advance whether  $\mathcal{A}$  uses a secret key  $\text{sk}_0$  or  $\text{sk}_1$ . Hence, we must define four auxiliary games  $\mathcal{G}_{ij}$  where  $\mathcal{G}_{ij}$  denotes a game  $\mathcal{G}_i$  that is halted with  $\perp$  if  $q \neq j$ . As

$$\Pr[\mathcal{G}_0^{\mathcal{A}} = 1] = \Pr[\mathcal{G}_0^{\mathcal{A}} = 1 \wedge q = 0] + \Pr[\mathcal{G}_0^{\mathcal{A}} = 1 \wedge q = 1] \quad , \quad (4.14)$$

$$\Pr[\mathcal{G}_1^{\mathcal{A}} = 1] = \Pr[\mathcal{G}_1^{\mathcal{A}} = 1 \wedge q = 0] + \Pr[\mathcal{G}_1^{\mathcal{A}} = 1 \wedge q = 1] \quad , \quad (4.15)$$

the straightforward application of the triangle inequality yields

$$\text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) \leq \text{cd}_*^t(\mathcal{G}_{00}, \mathcal{G}_{10}) + \text{cd}_*^t(\mathcal{G}_{01}, \mathcal{G}_{11}) \quad . \quad (4.16)$$

The analysis simplifies as games  $\mathcal{G}_{00}, \mathcal{G}_{10}$  do not use  $\text{sk}_1$  and  $\mathcal{G}_{01}, \mathcal{G}_{11}$  do not use  $\text{sk}_0$ . Consequently, we can now use IND-CPA transformation for both games:

$$\mathcal{G}_{00} \xrightarrow[\varepsilon]{\text{IND-CPA}} \mathcal{G}_{10} \quad \text{and} \quad \mathcal{G}_{01} \xrightarrow[\varepsilon]{\text{IND-CPA}} \mathcal{G}_{11} \quad . \quad (4.17)$$

Again, if the cryptosystem is  $(t, \varepsilon)$ -IND-CPA secure, there exists  $t_1 = t - O(1)$  such that  $\text{cd}_*^{t_1}(\mathcal{G}_{00}, \mathcal{G}_{10}) \leq \varepsilon$ ,  $\text{cd}_*^{t_1}(\mathcal{G}_{01}, \mathcal{G}_{11}) \leq \varepsilon$  and thus  $\text{cd}_*^{t_1}(\mathcal{G}_0, \mathcal{G}_1) \leq 2\varepsilon$ .

This horizon-splitting technique can be generalised to handle any finite set of exhaustive but mutually exclusive hypotheses  $\mathcal{H}_1, \dots, \mathcal{H}_n$  about the interaction pattern in the game  $\mathcal{G}$ . Namely, we can define narrowings of the game  $\mathcal{G}$ :

$$\mathcal{G}^{\mathcal{A}}|_{\mathcal{H}_i} = \begin{cases} \mathcal{G}^{\mathcal{A}}, & \text{if } \mathcal{H}_i \text{ holds} \quad , \\ \perp, & \text{otherwise} \quad . \end{cases} \quad (4.18)$$

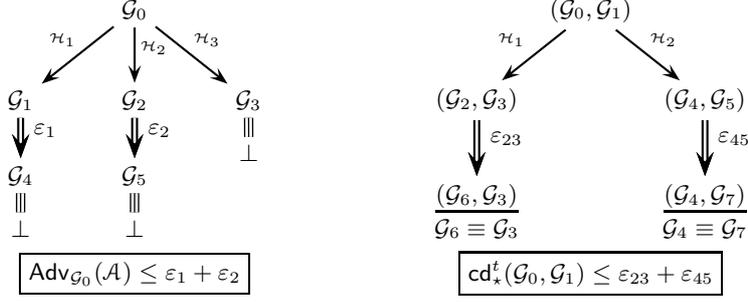


Figure 4.7: Game trees together with the corresponding bounds. Simple arrows represent horizon-splitting and double arrows represent reduction schemata.

where the verification of  $\mathcal{H}_i$  is done by the challenger. Now, the ordinary triangle inequality induces two tight splitting bounds

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \text{Adv}_{\mathcal{G}|\mathcal{H}_1}(\mathcal{A}) + \cdots + \text{Adv}_{\mathcal{G}|\mathcal{H}_n}(\mathcal{A}) , \quad (4.19)$$

$$\text{cd}_{\star}^t(\mathcal{G}_0, \mathcal{G}_1) \leq \text{cd}_{\star}^t(\mathcal{G}_0|\mathcal{H}_1, \mathcal{G}_1|\mathcal{H}_1) + \cdots + \text{cd}_{\star}^t(\mathcal{G}_0|\mathcal{H}_n, \mathcal{G}_1|\mathcal{H}_n) . \quad (4.20)$$

Horizon-splitting technique can be used whenever no elementary transformation is applicable due to the unknown behaviour of  $\mathcal{A}$ . In a sense, splitting bounds are more detailed analogs of the formal derivation rule

$$\frac{\Gamma, F \vdash A \quad \Gamma, G \vdash A}{\Gamma, F \vee G \vdash A} \quad (4.21)$$

in logic that allows us to strengthen the assumptions by splitting the proof into several branches. In our case, we get a game tree instead of a proof tree.

To summarise, we can bound the advantage  $\text{Adv}_{\mathcal{G}}(\mathcal{A})$  by constructing a game tree such that all leaf level games  $\mathcal{G}_i$  have trivial bounds  $\text{Adv}_{\mathcal{G}_i}(\mathcal{A}) \leq \varepsilon_i$ , since we can use triangle inequalities (4.13), (3.23) and splitting bounds (4.19)–(4.20) to estimate  $\text{Adv}_{\mathcal{G}}(\mathcal{A})$ . Moreover, we can find a tight upper bound on the running times by aggregating all time constraints corresponding to the edges of the game tree. Similarly, we can bound a computational distance  $\text{cd}_{\star}^t(\mathcal{G}_0, \mathcal{G}_1)$  but the corresponding proof tree consists of game pairs. Namely, the pair  $(\mathcal{G}_0, \mathcal{G}_1)$  is a root node and we can derive new nodes by applying reductions schemata to individual components of a pair, or alternatively create several siblings by using the horizon-splitting technique. Again, if all leaf level nodes  $(\mathcal{G}_i, \mathcal{G}_j)$  are bounded  $\text{cd}_{\star}^t(\mathcal{G}_i, \mathcal{G}_j) \leq \varepsilon_{ij}$ , then we can use inequalities (3.23) and (4.20) to compute the final bound  $\text{cd}_{\star}^t(\mathcal{G}_i, \mathcal{G}_j)$ . See Fig. 4.7 for illustrative examples. Note that such a proof technique is sound and complete. More formally, if we cannot represent a valid handcrafted proof as a game tree, then we can view the proof itself as a new reduction schema and thus obtain a two-element game chain.

This approach is particularly useful when we have to derive technically complex proofs, since it is straightforward to apply and it is immune to argumentation flaws. At the same time, the corresponding proofs can be extremely boring to read, since they are quite long and mechanical. Moreover, it is often easier to construct reductions for specific games than to formalise respective reduction schemata  $(\mathcal{P}, \mathcal{T}, \mathcal{S})$ . Thus, it is common to compress proof trees by providing more complex handcrafted reductions after the proof is derived.

## 4.5 FORMAL VERIFICATION OF CRYPTOGRAPHIC PROOFS

The methodology outlined above has slowly evolved together with the concept of exact security. As the main emphasis has always been on practical results, the meta-analysis of proof methodology itself is rather recent. Only in the year 2004, Bellare, Rogaway and Shoup started to talk about game-playing proofs as a general methodology [BR04, Sho04]. Although some interesting proof methods, such as [LMMS98, Mau02], have been published before, none of them has claimed to be universal, as the approach [BR04] proposed by Bellare and Rogaway. Next, Halevi took a step further and suggested that cryptographers should build an automatic tool for generating, analysing and verifying the game-playing proofs [Hal05]. The first proposal CRYPTOVERIF [BP06] for such a formal proof system is now available. Notably, the authors have taken effort to transform the ordinary program analysis lingo into an adversary-centric description language, which is almost understandable without any prior preparation.

Despite the existence of CRYPTOVERIF and the corresponding extensive documentation [Bla06b, Bla07] and numerous other approaches that work in the asymptotic setting (e.g. [CV01, Bla01, Low97, MRST06]), the author still feels compelled to explain the high-level structure of such a formal proof system. Firstly, a majority of these systems are quite unnatural from a cryptographer's viewpoint, as they are formalised by logicians, who have put proof theoretical aspects in the first place. Secondly, all of them are rather limited as they utilise only reduction schemata and not the horizon-splitting technique. In fact, the linearity of proofs is one of the main reasons why current automatic provers are so successful. To be fair, we note that horizon splitting is not a well known technique. In fact, only a weak form of it has been semi-officially published by Dent [Den06]. Finally, thinking in terms of proof systems makes certain properties and limitations of conditional proofs more apparent.

In the following, we describe a proof system in which the emphasis is placed on cryptographic aspects. In particular, note that the derivation of proofs is not the main problem in cryptography, the actual problem lies in the verification of subtle details. Hence, the automatic derivation of proofs is not a key feature at all, although the system should handle mundane tasks in the proofs, such as verification of the preconditions for various reduction schemata.

**High-level description.** Our hypothetical computer environment CRYPROOF should be a proof system in a strict sense, i.e., it is just an interactive environment for proving cryptographic statements about advantages and computational distances of different games. Secondly, CRYPROOF should directly manipulate games that are given in challenger- or adversary-centric description style similar to [BR04, App. B]. Since the imperative programming paradigm is dominant in cryptography, all alternative description styles based on various forms of process calculi have an extremely high entrance cost.<sup>2</sup> Finally, CRYPROOF should do the automatic bookkeeping and output the resulting time-success profiles given a characterisation of basic primitives used in the construction.

Let us now briefly describe how one should use CRYPROOF. As explained in the previous section, any security proof is a game tree and the CRYPROOF envi-

---

<sup>2</sup>Repeated switching between different programming paradigms is annoying both from a technical and cognitive viewpoint. Engineers and cryptographers just do not do subtitles!

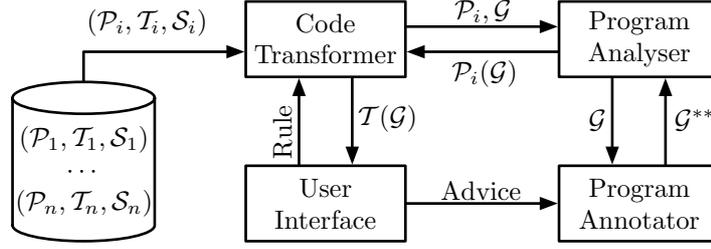


Figure 4.8: High-level description of the proof system CRYPROOF.

ronment should assure that a game tree is properly formed. Hence, a single step in CRYPROOF is either an application of a reduction schemata or a horizon-splitting step. Testing the validity of horizon-splitting step is more straightforward, as we must check that hypotheses  $\mathcal{H}_1, \dots, \mathcal{H}_m$  are mutually disjoint and exhaustive. The application of a reduction schema is more complex, see Fig. 4.8. Essentially, we must check that a precondition  $\mathcal{P}_i$  holds for the game  $\mathcal{G}$ . Since a precondition describes a non-trivial property of a program  $\mathcal{G}^A$ , the predicate  $\mathcal{P}_i(\mathcal{G})$  might be undecidable. Consequently, a user must be able to prove  $\mathcal{P}_i(\mathcal{G})$  by annotating the program  $\mathcal{G}$  when automatic methods fail.

Any formal annotation method that makes the verification of  $\mathcal{P}_i(\mathcal{G})$  easy is suitable for the task. Hoare logic is one of the best choices, since it resembles the way we actually reason about properties of a program. In a nutshell, Hoare logic is a systematic way to label the code statements with pre- and postconditions. For example, a labelling  $\{y \leq 5\} y \leftarrow y + 3 \{y \leq 8\}$  means that if  $y \leq 5$  before the statement  $y \leftarrow y + 3$  then  $y \leq 8$  afterwards. Notably, only a small language dependent set of derivation rules determines the labelling for the whole algorithm. Thus, one can prove the correctness of the labelling by giving the corresponding derivation tree. For more detailed discussion see [NN92].

Note that the challenger can always test whether  $\mathcal{P}_i(\mathcal{G})$  holds for any particular run. Let  $\mathcal{G}^*$  be the description of a modified game, where the challenger sets a flag  $\text{bad} \leftarrow 1$  if the precondition  $\mathcal{P}_i(\mathcal{G})$  does not hold. Then a valid Hoare triple  $\{\emptyset\} (\mathcal{G}^*)^A \{\text{bad} \neq 1\}$  proves that the condition  $\mathcal{P}_i$  holds for all runs of  $\mathcal{G}^A$ . Consequently, a valid annotation  $\mathcal{G}^{**}$  that proves  $\{\emptyset\} (\mathcal{G}^*)^A \{\text{bad} \neq 1\}$  is also a proof for  $\mathcal{P}_i(\mathcal{G})$ . Moreover, all provable properties of an algorithm are provable also in Hoare logic provided that one allows a sufficiently rich description language for the pre- and postconditions [NN92]. Therefore, the annotation  $\mathcal{G}^{**}$  exists if and only if the condition  $\mathcal{P}_i(\mathcal{G})$  is indeed provable.

All other operations, such as applying the transformations, constructing the proof trees and computing various bounds, are straightforward. Clearly, the database of reduction schemata  $\{(\mathcal{P}_i, \mathcal{T}_i, \mathcal{S}_i)\}$  is the central part of CRYPROOF, since it captures the domain knowledge. Also, it is likely to be incomplete at the beginning. Therefore, we must update it when handcrafted security proofs reveal missing reduction schemata, i.e., some proofs cannot be formalised in the CRYPROOF environment. The task of the system maintainer is to carefully verify handcrafted proofs to extract and generalise the corresponding reduction schema. The latter is the only non-engineering task in the long run.

**Axioms and deduction rules.** Normally, a domain of knowledge is formalised

by axioms. Indeed, all reduction schemata  $(\mathcal{P}_i, \mathcal{T}_i, \mathcal{S}_i)$  are axiomatic theorems. However, it is far easier to interpret them as deduction rules and postulate that one must apply reduction schemata until the leafs of game trees contain only pairs  $\mathcal{G}_i \equiv \mathcal{G}_j$  or single games  $\mathcal{G}_i \equiv \perp$ , see again Fig. 4.7.

Note that our proof system must have deduction rules for code simplifications that do not change the behaviour of  $\mathcal{G}^A$ , such as algebraic manipulations and dead code elimination. For advantage computations, we need a special WRAP rule that formalises the intuition that advantage cannot decrease if we push some computations from the honest challenger into the malicious adversary. Similarly, the computational distance cannot increase if some messages sent out by the challenger are dropped and never reach the adversary  $\mathcal{A}$ .

The second group of transformations is used for manipulating cryptographic primitives like  $\text{Enc}_{\text{pk}}(\cdot)$  or  $\text{Hash}(\cdot)$ . As such symbols represent unknown functions, we must finally eliminate them from a game  $\mathcal{G}$ , otherwise it is formally impossible to establish bounds on  $\text{Adv}_{\mathcal{G}}(\mathcal{A})$ . Therefore, for every legitimate usage of a primitive, there must be an elimination rule. Consider a cryptosystem  $(\text{Gen}, \text{Enc}, \text{Dec})$  as an example. If a game  $\mathcal{G}$  uses only the  $\text{Gen}$  symbol, then the WRAP rule is sufficient to remove  $\text{Gen}$ . If a game  $\mathcal{G}$  does not contain  $\text{Dec}_{\text{sk}}(\cdot)$  symbol, then IND-CPA and WRAP rule together are sufficient to eliminate  $\text{Gen}$  and  $\text{Enc}_{\text{pk}}(\cdot)$  symbols. For the  $\text{Dec}_{\text{sk}}(\cdot)$  symbol, there are three alternatives. The WRAP rule is sufficient if there are no  $\text{Enc}_{\text{pk}}(\cdot)$  symbols, since key generation together with decryption can be wrapped into  $\mathcal{A}$ . Secondly, in code lines  $c \leftarrow \text{Enc}_{\text{pk}}(m_1), \dots, m_2 \leftarrow \text{Dec}_{\text{sk}}(c)$ , we can simplify  $m_2 \leftarrow m_1$ . In the remaining cases, we need IND-CCA2 transformation, as  $\mathcal{A}$  can alter ciphertexts.

The third class of transformations is used in the probability analysis after all cryptographic primitives are removed from the games. Such transformations are extensively used in primitive design, where the elimination of all cryptographic primitives can create difficult combinatorial problems. Notably, most of these reductions are based on the BAD reduction schema, see [BR04]. In a nutshell, we can substitute any code block  $\text{seg}_0$  with a block  $\text{bad} \leftarrow 1; \text{seg}_1$  provided that  $\text{seg}_0$  is reached with low probability. Let  $\mathcal{G}_0^A(\omega)$  and  $\mathcal{G}_1^A(\omega)$  denote random variables that correspond to the transformation  $\mathcal{G}_0 \xrightarrow[\varepsilon]{\text{BAD}} \mathcal{G}_1$ , where  $\omega = (\omega_{\mathcal{A}}, \omega_{\mathcal{G}})$  consists of all random coins used by the adversary and the challenger. Then values  $\mathcal{G}_0^A(\omega)$  and  $\mathcal{G}_1^A(\omega)$  can differ only if  $\text{bad} = 1$  and we obtain

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A}) \leq \Pr[\omega \leftarrow \Omega_* : \mathcal{G}_0^A(\omega) \neq \mathcal{G}_1^A(\omega)] \leq \Pr[\omega \leftarrow \Omega_* : \text{bad} = 1] .$$

To bound  $\Pr[\text{bad} = 1]$ , we have to specify a set  $\Omega_{\Delta} \supseteq \{\omega \in \Omega_* : \text{bad} = 1\}$  and an estimate  $\Pr[\Omega_{\Delta}] \leq \varepsilon_{\Delta}$ . Now if we omit the probability calculations, then a valid annotation of the Hoare triple  $\{\omega \notin \Omega_{\Delta}\} \mathcal{G}_1^A(\omega) \{\text{bad} = 0\}$  is sufficient. Hence, we can formally verify combinatorial transformations, which seem to be the major hurdle in symmetric primitive design [Mau02, Hal05].

**Relativised models.** A conditional security proof is meaningful only if there exists a plausible instantiation of abstract primitives. Since direct proofs are practically out of reach, we must indirectly verify that various abstract demands for the primitive are not contradictory. More precisely, we need a computational model, where all instantiations of all basic primitives do exist and are provably secure. As we use *black-box specification* for all primitives, it is straightforward to build such a model by employing external oracles.

For concreteness, we construct a model for IND-CPA secure cryptosystem. Assume that an external oracle  $\mathcal{O}$  provides replies to Gen, Enc and Dec queries. To serve the  $i^{\text{th}}$  query Gen, the oracle draws  $r_i \leftarrow \{0, 1\}^n$ , stores  $(i, r_i)$  and outputs a key pair  $((i, r_i), i)$ . To serve  $\text{Enc}(i, m)$ , the oracle stores  $m$  into the first open slot of an array  $(m_{ij})_{j=1}^{\infty}$  and returns the corresponding index number  $j_0$ . Let  $\text{Dec}((i, r), c) = \perp$  if no recorded secret key coincides with  $(i, r)$  and  $\text{Dec}((i, r), c) = m_{ic}$  otherwise. Note that the cryptosystem is functional and  $(t \cdot 2^{-n}, t)$ -IND-CPA secure, since any  $t$ -time adversary can try out only  $t$  potential secret keys and the ciphertext is independent from a message.

**Limitations of conditional proofs.** It is not trivial to guess whether a conditional proof provides optimal security bounds or whether a security claim is provable at all. To solve such conundrums, we can provide counterexamples for establishing that we cannot have a conditional proof unless we add new assumptions. Commonly, these proofs describe a relativised world, where all basic primitives exist but a new construction is insecure. As a result, there are no valid security proofs provided that the proof system itself is consistent. However, these *oracle separation* results do not eliminate the possibility of direct proofs, as the additional usage of Peano axioms might reveal additional relations between abstract security properties that make these proofs possible.

For stronger separation results, we need to construct a counterexample from the basic primitives that are assumed to exist. Namely, we must construct a secondary set of weakened primitives that explicitly invalidates the hypothetical security claim. Such a *black-box separation* proves that there are no valid security proofs as long as the initial set of basic primitives exists and the proof system itself remains consistent. The separation itself can be as artificial as needed. For example, we can prove that IND-CPA security is insufficient for IND-CCA2 security by defining a new encryption rule  $\bar{\text{Enc}}_{\text{pk}}(m) = b \parallel \text{Enc}_{\text{pk}}(m)$  where  $b \leftarrow \{0, 1\}$ . To decrypt a message, one just has to omit the first bit. Clearly, the new cryptosystem is  $(t, \varepsilon)$ -IND-CPA secure whenever the original cryptosystem is  $(t + O(1), \varepsilon)$ -IND-CPA secure, but not IND-CCA2 secure, as we can flip the first bit of the encryption to fool the decryption oracle.

As there are many sub-optimal ways to prove security, separation techniques are often employed to show that obtained security guarantees are tight or at least near-optimal. At the same time, such separation results do not guarantee that bounds remain optimal if we take a more fine-grained view on the construction and break some basic primitives into smaller ones. Neither do these separation results prove that some constructions are inherently insecure, sometimes they just indicate underspecification of security premises.

## 5 SECURITY OF INTERACTIVE COMPUTATIONS

For obvious reasons, we can rigorously analyse only the security of well-specified distributed computations. Such well-specified distributed algorithms are commonly referenced as *protocols*. Security analysis of protocols is one of the main tasks in cryptography. Essentially, we can talk about security in two different contexts. We can analyse the security of a protocol in a *stand-alone setting*, where participants do not execute any other protocols. Alternatively, we can consider the security of a protocol in a wider computational context, where several protocols are executed to achieve more complex computational goals. In this chapter, we consider only stand-alone security.

Superficially, the stand-alone setting is quite restrictive, since protocol outputs cannot be processed further. In Chapter 7, we show that this restriction is purely formal and stand-alone security is sufficient to achieve sequential composability. In layman's terms, a protocol remains secure if no other protocols are executed in parallel. For many settings, such a requirement is naturally forced by the design of the system and thus the provided security level is sufficient. We discuss these issues more thoroughly in the following chapters, as even the stand-alone security model is complex enough to confuse non-experts.

The number of potential security models is humongous even in the stand-alone setting. There are literally hundreds of different security models that characterise various practical settings. Although only few of them are well studied, the others are equally meaningful. Therefore, we concentrate on the real versus ideal world paradigm that forms the core of all security models.

The real versus ideal world concept itself is easy to formalise—a protocol instance is secure if it is sufficiently similar to an ideal implementation. However, it is somewhat nontrivial to describe what the similarity between real and ideal world exactly implies or what happens if we change some minor details in the formalism. Thus, we start with basic consistency principles that must hold for any reasonable formalisation. Next, we consider what security levels are achievable in the ideal setting. Although the latter may seem a rather absurd question to ask as we cannot do any better, it is actually one of the most important ones. If the risks caused by joining an ideal world protocol are not rewarded by the potential gains, then the whole task is hopeless from the start and there is no reason to design a cryptographic solution at all. Secondly, the ideal world behaviour also reveals what kind of properties must be preserved by the correspondence between real and ideal world adversaries.

As a result, we can justify all details of the ideal versus real world approach from a game theoretical viewpoint. The latter provides stronger validity guarantees, since all security definitions that contradict or ignore economical principles are meaningless in practice. Secondly, only a deep understanding of desired security goals provides a justified way to choose a proper security model. Essentially, there are four basic levels of security ranging from input-privacy to complete security and all of them are covered in this chapter. More advanced topics, like asymptotic security and setup assumptions, are left out and are treated separately in remaining chapters.

## 5.1 FORMAL REQUIREMENTS TO SECURITY DEFINITIONS

Before we actually start formalising various security goals, it is wise to contemplate what software and hardware designers mean by the terms ‘attack’ and ‘security’ and what kind of results are expected from cryptographers. These questions are often considered trivial and thus ignored. The ignorance is often rewarded by aesthetically pleasing formalisations and beautiful theories. But at the same time, these results can be cumbersome, not to say completely unrelated to the demands of everyday life. Beautiful theories that are touching the limits of mathematical reasoning are not something to be ashamed of. However, we choose a viewpoint that is as close to practice as possible.

Obviously, all security definitions must be well defined and non-contradictory with itself. Briefly, a cryptographic construction either meets the security definition or not and alternative descriptions of a same construction are either all secure or all insecure. This property is known as internal consistency.

However, the definition has to be appropriate also for practical settings, or otherwise we have hopelessly departed from reality. To assure that a security definition has a clear interpretation in practice, we postulate four restrictions to capture external consistency. First, the effect of an attack must be observable and unambiguously detectable, as emphasised by the restrictions (C1)–(C2). Second, a security notion must be defined as a resistance level against potential attacks, as emphasised by the remaining restrictions (C3)–(C4).

- (C1) The effect of a successful attack must be observable. If an attack induces changes that manifest only in the internal retrospection of a participant and not in any observable outcome, then the attack is not successful.
- (C2) For each protocol run and each attack against the protocol, it must be possible to determine whether the attack was successful or not. More formally, the overall success of an adversarial strategy  $\mathcal{A}$  is quantified in terms of an appropriate game  $\mathcal{G}$  such that  $\mathcal{G}^{\mathcal{A}} \in \{0, 1, \perp\}$  and the advantage  $\Pr[\mathcal{G}^{\mathcal{A}} = 1]$  is the corresponding quantitative measure.
- (C3) A construction is secure if no plausible adversarial algorithm is successful enough. For example, a construction is  $(t, \varepsilon)$ -secure if the advantage  $\Pr[\mathcal{G}^{\mathcal{A}} = 1] \leq \varepsilon$ , for all plausible  $t$ -time adversaries  $\mathcal{A}$ .
- (C4) Adding restrictions to attack algorithms can only increase the security level. For example, if a construction is  $(t_1, \varepsilon)$ -secure and not  $(t_2, \varepsilon)$ -secure for  $t_1 > t_2$ , the corresponding success criterion is inconsistent.

Let us emphasise some details before going any further. First, it is really impossible to detect how a person or an algorithm internally perceives the situation when there is no measurable outcome. Moreover, we as designers are often interested only in certain behavioural aspects. For example, the output of a poker automate does not have to be truly pseudorandom. A player may even register certain regularities, but the latter is completely irrelevant as long as it does not change his or her gaming behaviour or winning odds.

Secondly, the external consistency requirements are crucial only in the context of practical security, where one has to make economical decisions based on cryptographic results. Consequently, cryptographic definitions may violate some

of the postulates (C1)–(C4), as long as they are used as tools to achieve externally consistent security goals. For example, the original definition of universal composability [Can01] violates assumptions (C2)–(C3), and at first glance some asymptotic definitions of statistical security [Gol04, Can00a] contradict postulates (C3)–(C4). For clarity, we note that universal composability has also an externally consistent formulation [Lin03b, Can00b], and that the problems with statistical security are caused by ambiguities in the interpretation, see Section 7.6 and Section 6.2. Nevertheless, one can view the external consistency postulates as a litmus test for detecting possible errors in the formalisations.

Thirdly, a bound of the total running time is often a too coarse measure for defining a class of plausible adversarial strategies. For most protocols, an adversary must provide replies to the other participants in a certain time frame and it is justified to quantify online and offline complexity of the attack. Now, if we add some additional constraints, then the conditions (C3)–(C4) must be modified accordingly to capture all monotonicity requirements.

## 5.2 SECURITY OF IDEALISED COMPUTATIONS

Suitability of a cryptographic construction can be viewed as any other design problem: we have to state some clear objectives and then assess the construction in respect to them. For example, we may state that an e-voting scheme is secure if one cannot submit incorrect votes, valid votes are correctly counted, and it is impossible to disclose the choice made by an individual voter. However, such a specification is incomplete, as it does not rule out other possible abuses. For example, a voter may still be able to sell her vote, if she can disclose her vote in an indisputable way to a potential buyer. We can add this restriction to the black-list of malicious behavioural patterns but we can never be sure that the list is complete. Also, it may be difficult to verify that such a black-list specification is internally consistent, i.e., does not contain non-resolvable conflicts between security goals. Therefore, a white-list based specification that explicitly states what behavioural patterns are plausible is a far better alternative.

Let us start with some simplifying conventions. First, let symbols  $\mathcal{P}_1, \dots, \mathcal{P}_n$  always denote the participants of a protocol. Second, a protocol must implement some, possibly randomised, functionality  $f_1, \dots, f_n$  in order to be useful, i.e., each party should obtain  $y_i = f_i(x_1, \dots, x_n)$ , where  $x_1, \dots, x_n$  are the inputs of all participants. The computational process itself can be either non-adaptive or adaptive. For non-adaptive protocols, the functionality  $f_1, \dots, f_n$  is known ahead, whereas  $f_i$  is determined dynamically during an adaptive protocol. In this chapter, we consider only non-adaptive computations. Adaptive protocols are normally composed from smaller non-adaptive basic blocks and thus it makes sense to consider them together with composability in Chapter 7.

Additionally, we have to resolve the question whether the inputs  $x_1, \dots, x_n$  are automatically deleted after the completion of a protocol or not. Both choices are equally expressive but preserving the inputs is technically cleaner and looks more natural. Therefore, we require that each non-corrupted node  $\mathcal{P}_i$  outputs  $z_i = (x_i, y_i)$  at the end of the protocol. Such requirement makes it straightforward to consider the security in the scenarios, where the output  $z_i$  is further

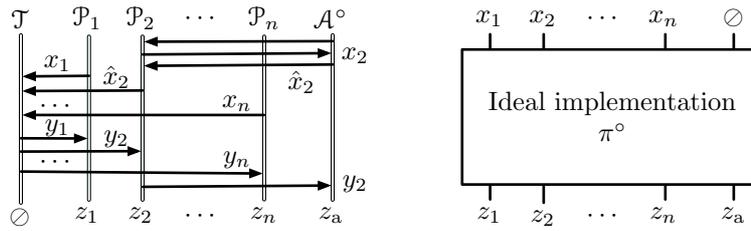


Figure 5.1: Ideal implementation of a protocol  $\pi$ . An example attack on the left and a schematic description as an abstract computational block on the right.

processed by  $\mathcal{P}_i$  locally, or used as an input to another protocol.

**Idealised computational process.** Note that a protocol must reveal some information about the inputs as long as it implements a desired functionality  $f_1, \dots, f_n$ . Namely, if parties  $\mathcal{P}_1, \dots, \mathcal{P}_n$  run a protocol  $\pi$ , then each of them must learn  $f_i(x_1, \dots, x_n)$ . Moreover, a malicious party  $\mathcal{P}_i$  can always change the outcome of  $\mathcal{P}_j$  by choosing a different input  $\hat{x}_i$  instead of  $x_i$ . Finally, a party  $\mathcal{P}_i$  may decide to halt in the middle of the protocol or not to participate at all. Obviously, no cryptographic countermeasure can avoid these actions.

Cryptographers use a concept of ideal implementation to model such unavoidable attacks. The ideal implementation  $\pi^\circ$  of a protocol is defined using a trusted third party  $\mathcal{T}$  that does all computations on behalf of the others, see Fig. 5.1. That is, all participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  first submit their inputs  $x_1, \dots, x_n$  securely to the trusted third party  $\mathcal{T}$ . Next, the trusted third party  $\mathcal{T}$  computes the desired outcomes  $y_i \leftarrow f_i(x_1, \dots, x_n)$  and sends them securely back to the participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$ . In particular, if a party  $\mathcal{P}_i$  does not submit his or her input  $x_i$ , the protocol is halted and  $\mathcal{T}$  sends  $\perp$  to all participants. As a result, either all participants will get their desired outputs or neither of them will.

Alternatively, we can fix a schedule how  $\mathcal{T}$  must deliver the replies and allow participants to stop  $\mathcal{T}$ . More formally, a participant  $\mathcal{P}_i$  must send zero-one verdict to  $\mathcal{T}$  after receiving  $y_i$ . If a participant  $\mathcal{P}_i$  sends 1,  $\mathcal{T}$  continues as usual, otherwise  $\mathcal{T}$  halts and sends  $\perp$  instead of all remaining replies  $y_j$ .

The second idealised model does not guarantee *fairness*, since a malicious party  $\mathcal{P}_i$  can force honest parties to compute  $f_i(x_1, \dots, x_n)$  and halt computations before the others obtain their outputs. Clearly, it is easier to design protocols without fairness. Moreover, fundamental limits already outlined by Cleve [Cle86] imply that fairness is unachievable for two-party protocols and protocols with honest minority. Although certain solutions like gradual release of secrets [LMR83, BG89, GL90] and use of external referees [ASW98, CC00] can alleviate the problem, none of them is universally applicable.

**Tolerated adversarial behaviour.** In many settings, participants can form coalitions to achieve their malicious goals. We model such behaviour by considering an ideal world adversary  $\mathcal{A}^\circ$  which can corrupt some of the nodes  $\mathcal{P}_1, \dots, \mathcal{P}_n$ , see again Fig. 5.1. The set of possible coalitions can be specified by listing all possible subsets of parties that can be corrupted. The corresponding set of sets  $\mathcal{A}$  is called an *adversary structure*. Obviously, if it is possible to corrupt a subset

$A \in \mathcal{A}$ , it is also possible to corrupt a subset  $B \subseteq A$ . Hence, we can always assume that the adversarial structure  $\mathcal{A}$  is monotone.

Exact details about the nature of a plausible corruption determine the maximal achievable damage. Adversarial behaviour can be static, adaptive or mobile. A *static adversary* corrupts all participants already before the beginning of the protocol. An *adaptive adversary* corrupts parties adaptively, based on the acquired information; a *mobile adversary* can additionally retreat from corrupted nodes. A mobile adversary respects the adversarial structure  $\mathcal{A}$  if at any moment in time, the adversary controls a subset of participants that belongs to  $\mathcal{A}$ . We emphasise here that adaptive and mobile adversaries can corrupt participants also after protocol values  $y_i$  are sent out by  $\mathcal{T}$ .

In a sense, the mobile adversary models the reality most precisely in large infrastructures, whereas static and adaptive adversaries are more suited for small well-protected networks. Two-party protocols always have static adversaries, as the adversarial structure is simplistic  $\mathcal{A} = \{\{\mathcal{P}_1\}, \{\mathcal{P}_2\}\}$ .

**Non-standard ideal world models.** Sometimes all parties  $\mathcal{P}_1, \dots, \mathcal{P}_n$  are guaranteed to be honest in the real world and the ideal adversary  $\mathcal{A}^\circ$  represents the effects of malicious outsiders. For example, key agreement protocols [BR93a, PV06b, LN06] try to resolve problems caused by malicious message transmission. In such cases, it is more natural to include  $\mathcal{A}^\circ$  as a direct participant in the ideal computation instead of introducing a dummy corrupted party  $\mathcal{P}_0$ .

There are many similar minor details that can be varied to express design goals more succinctly. For example, the idealised computational process for adaptive tasks may have more than two rounds. In all these non-standard cases, the ideal world model can still be decomposed into a model of idealised computational behaviour and a model of tolerable adversarial behaviour.

**Highest achievable security level.** An ideal world implementation itself does not guarantee privacy of inputs nor correctness of outputs. The model just shows, which security goals are achievable for all protocols that correctly implement the desired functionality. For example, if two parties both want to obtain  $y = x_1 + x_2$ , then the output  $y$  reveals the opponent's input and thus privacy of inputs cannot be preserved. Therefore, it is important to understand what kind of security guarantees are achievable at all. Although we cannot answer these questions without making additional assumptions about the adversarial behaviour, it is still an important aspect to consider. If the ideal implementation fails to provide a desired security level, there is no reason to design the corresponding cryptographic protocols. Secondly, a proper understanding of ideal world attacks provides an important insight into what kind of correspondence between ideal and real world implementations is needed.

Recall that any externally consistent security goal can be formalised as a game, see Section 5.1. Although protocol inputs are normally fixed, it is more instructive to first consider a game that measures average-case security.

$$\mathcal{G}^{\mathcal{A}^\circ} \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathcal{D} \\ \mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ}(\mathbf{x}) \\ \text{return } \mathcal{B}(\mathbf{z}_{\text{obs}}) \end{array} \right.$$

That is, the challenger first draws inputs  $\mathbf{x} = (x_1, \dots, x_n)$  from an input distribution  $\mathcal{D}$  and then runs a sub-game  $\mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ}(\mathbf{x})$  to simulate the ideal world

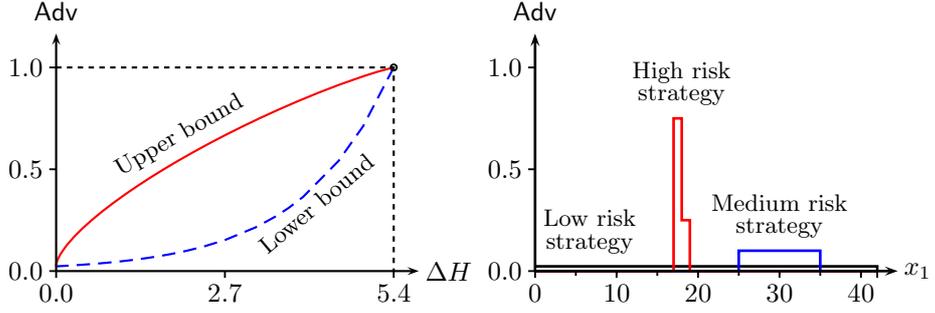


Figure 5.2: Analysis of ideal addition protocol for an attack target  $x_1 + \hat{x}_2 = 42$ . Dependence between the prior knowledge and the advantage on the left and dependence between the advantage and the true value of  $x_1$  on the right.

attack, and finally evaluates a predicate  $\mathcal{B}(\cdot)$  on the set of observable outcomes  $\mathbf{z}_{\text{obs}} = (z_1, \dots, z_n, z_a)$  to determine whether an attack was successful or not. Now the maximal achievable advantage depends on the input distribution  $\mathfrak{D}$  and on the maximal running time of  $\mathcal{A}^\circ$ . Let us first consider the uniform distribution  $\mathfrak{D}_u$  where all plausible inputs are equiprobable. Then

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ) = \frac{1}{|\mathfrak{D}|} \cdot \sum_{\mathbf{x} \in \mathfrak{D}} \text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x}) , \quad (5.1)$$

where

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x}) = \Pr [\mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{id-attk}}^{\mathcal{A}^\circ}(\mathbf{x}) : \mathcal{B}(\mathbf{z}_{\text{obs}}) = 1] \quad (5.2)$$

quantifies the success probability for a fixed input  $\mathbf{x}$ .

As a concrete example, consider a scenario, where a cheating party  $\mathcal{P}_2$  wins if the output of ideal addition protocol is 42. For clarity, assume also that the trusted third party  $\mathcal{T}$  provides outputs to participants only if submitted inputs  $x_1, \hat{x}_2$  are in the range  $\{0, 1, 2, \dots, 42\}$ . As the output of  $\mathcal{P}_1$  is  $z_1 = (x_1, y_1)$ , the security goal  $\mathcal{B}(z_1, z_2, z_a) = 1$  iff  $y_1 = x_1 + \hat{x}_2 = 42$  is indeed legitimate. Now the right sub-figure of Fig. 5.2 depicts graphically the dependence of  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | x_1)$  and shows clearly that different strategies can lead to different trade-offs between maximal and minimal success. A sharp and peaky *risk profile*  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x})$  indicates that the strategy is risky, since the success probability is high for a few inputs. As the input  $\mathbf{x}$  will rarely fall to the high success region, the adversary executes an all-or-noting attack strategy. Conservative uniformly flat risk profiles minimise the potential losses with the cost of decreasing the maximal possible winning probability. That is, it is possible to win the game for each input  $\mathbf{x}$ , but the corresponding success probability is equally low.

As the time-success profile  $\varepsilon(t)$  w.r.t.  $\mathfrak{D}_u$  limits the area under a risk profile, it also determines the basic properties of various trade-offs. In particular, no  $t$ -time strategy  $\mathcal{A}^\circ$  can guarantee  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x}) > \varepsilon(t)$  for all  $\mathbf{x}$  and  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x}) \approx \varepsilon(t)$  for conservative strategies. Moreover,  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ | \mathbf{x}) \geq c \cdot \varepsilon(t)$  only for  $\frac{1}{c}$  fraction of inputs and thus achieving high  $c$  values means taking a huge risk.

Situation can change drastically if an adversary has additional prior information about the inputs. Formally, such prior information is modelled as a *subjective* input distribution  $\mathfrak{D}$  that characterises the adversaries *beliefs* on the likelihood of various inputs. For example, the knowledge  $7 \leq x_1 \leq 17$  is represented

by the uniform distribution over  $\{7, \dots, 17\}$ . As the distribution  $\mathfrak{D}$  is fixed, the adversary can find an optimal strategy  $\mathcal{A}^\circ$  that maximises  $\text{Adv}_{\mathcal{G}}(\mathcal{A}^\circ)$  or evaluate the subjective risk of missing a  $\varepsilon_0$ -success region

$$\Pr[\mathbf{x} \leftarrow \mathfrak{D} : \text{Adv}(\mathcal{A}^\circ | \mathbf{x}) \leq \varepsilon_0] . \quad (5.3)$$

Again, the time-success profile  $\varepsilon(t)$  w.r.t.  $\mathfrak{D}$  describes basic choices for trade-offs between risk and success. We emphasise here that using prior information leads to sharper risk profiles and consequently a potential attacker may seriously underperform if the used information is fraudulent. In other words, the potential attacker must really believe in the validity of information.

For many tasks, we can even quantify the direct effect of prior information to success probability. Namely, we can measure the information content in bits as a difference between maximal and actual Shannon entropy of variables unknown to the adversary, and compute the corresponding success bounds, see Fig. 5.2. Similarly, we can quantify the inevitable privacy loss of a protocol as

$$\Delta H = H(\mathcal{X}_{\text{prior}}) - H(\mathcal{X}_{\text{post}}) , \quad (5.4)$$

where  $\mathcal{X}_{\text{prior}}$  is the prior and  $\mathcal{X}_{\text{post}}$  is the posterior distribution of unknown variables and  $H(\cdot)$  denotes Shannon entropy. Note that such bounds characterise only average properties and may underestimate the privacy loss for concrete inputs. Nevertheless, if an ideal implementation causes a huge privacy loss measured in hundreds or thousands of bits, then it might be acceptable to have a leakage of few extra bits in the real protocol to gain higher efficiency.

So far we have considered the situation from the adversarial viewpoint. The choices of honest parties are even more difficult, as the expected attack strategy depends on the risk tolerance and the prior knowledge of a potential adversary. Therefore, one must make even more subjective assumptions about adversarial behaviour. In principle, such choices can be summarised as a single randomised adversarial strategy  $\mathcal{A}^\circ$ . Consequently, we can use the corresponding security profiles to make justified economical decisions about potential losses. In practice, one often makes the decision based on a hunch or has no choice at all. Still, it is important to know that the basic security and the game theoretical properties of the ideal model are captured by time-success and risk profiles.

### 5.3 THE REAL VERSUS IDEAL WORLD PARADIGM

Evidently, a protocol provides the highest achievable security level if it satisfies all externally consistent security goals that are achievable in the ideal world. Such a reductionist view is quite old and can be traced back to the early works of Beaver, Micali and Rogaway [Bea91b, MR91b]. Our approach is based on the formalisation given by Canetti and Goldreich [Can00a, Gol04]. However, there are minor differences between our approach and the one pursued by Canetti and Goldreich. We discuss these matters explicitly at the end of this section.

**Formal definition of proximity.** As a starting point, recall that any externally consistent security goal can be formalised as a game between an adversary and a challenger. More precisely, such a game can be split into online and offline phases. The online phase of a game models the attack against the protocol, where

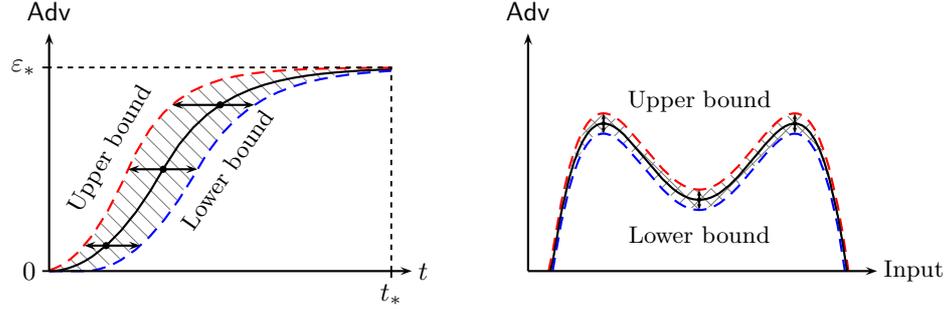


Figure 5.3: Correspondence between ideal and real world models. A real world time-success profile can be approximated with the corresponding ideal world profile. Similarly, the risk profile of  $\mathcal{A}^\circ$  induces bounds for the risk profile of  $\mathcal{A}$ .

all participants reach a final observable outcome  $z_i$ . In the subsequent offline phase, the challenger decides, based on the observable outcomes  $\mathbf{z}_{\text{obs}}$  whether the attack was successful or not. Hence, the offline phase can be formalised as a predicate  $\mathcal{B}(\cdot)$  that is independent of the implementation details and thus we can compare ideal and real world implementations.

More formally, let  $\mathcal{G}_{\text{re-atk}}$  and  $\mathcal{G}_{\text{id-atk}}$  denote sub-games that model the execution of a real and an ideal protocol. Let  $\mathcal{A}$  and  $\mathcal{A}^\circ$  be the corresponding real and ideal world adversaries. Then, for any input distribution  $\mathcal{D}$ , we can compare the advantages of the corresponding security games:

$$\begin{array}{l} \mathcal{G}_{\text{real}}^{\mathcal{A}} \\ \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathcal{D} \\ \mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{re-atk}}^{\mathcal{A}}(\mathbf{x}) \\ \text{return } \mathcal{B}(\mathbf{z}_{\text{obs}}) \end{array} \right. \end{array} \quad \begin{array}{l} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \\ \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathcal{D} \\ \mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ}(\mathbf{x}) \\ \text{return } \mathcal{B}(\mathbf{z}_{\text{obs}}) \end{array} \right. \end{array}$$

The sub-game  $\mathcal{G}_{\text{id-atk}}(\mathbf{x})$  is played as follows. The challenger runs the participants  $\mathcal{P}_1, \dots, \mathcal{P}_n, \mathcal{T}$  with inputs  $\mathbf{x} = (x_1, \dots, x_n)$  and interacts with the adversary  $\mathcal{A}^\circ$  according to the specification of the ideal world. In particular, the challenger immediately halts the game if  $\mathcal{A}^\circ$  violates the description of tolerable adversarial behaviour. Also, the challenger gives control over a participant  $\mathcal{P}_i$  to the adversary  $\mathcal{A}^\circ$  as soon as  $\mathcal{A}^\circ$  corrupts  $\mathcal{P}_i$ . Finally, all participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  and  $\mathcal{A}^\circ$  halt with some output values  $z_1, \dots, z_n$  and  $z_a$  and the corresponding vector  $\mathbf{z}_{\text{obs}} = (z_1, \dots, z_n, z_a)$  is returned as an output of  $\mathcal{G}_{\text{id-atk}}(\mathbf{x})$ .

The specification of the sub-game  $\mathcal{G}_{\text{re-atk}}(\mathbf{x})$  is similar. The challenger runs the participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  with inputs  $\mathbf{x} = (x_1, \dots, x_n)$  and interacts with the adversary  $\mathcal{A}$  according to the description of the protocol. Again, the challenger halts the game immediately if  $\mathcal{A}$  violates the description of tolerable adversarial behaviour. Finally, all participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  and the adversary  $\mathcal{A}$  halt and the vector of observable outputs  $\mathbf{z}_{\text{obs}}$  is returned as an output of  $\mathcal{G}_{\text{re-atk}}(\mathbf{x})$ .

Now consider a situation where, for a fixed security goal  $\mathcal{B}(\cdot)$  and for any real world adversary  $\mathcal{A}$ , there exists a comparable ideal world adversary  $\mathcal{A}^\circ$  such that the corresponding advantages are negligibly close

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon \quad (5.5)$$

for any input distribution  $\mathcal{D}$ . Then the corresponding time-success profiles in the real and ideal world are also close as illustrated in Fig. 5.3. More precisely,

for any fixed input distribution and a time bound, we can execute the corresponding optimal ideal world attack with a constant overhead in the real world. The latter gives a lower bound for the time-success profile. The the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  that satisfies the inequality (5.5) explicitly determines the corresponding upper bound. Moreover, as the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  preserves closeness for all input distributions, then for any real world attack  $\mathcal{A}$ , there exists a generic attack  $\mathcal{A}^\circ$  with a similar risk profile and comparable running time that is applicable against any protocol that implements the functionality. In other words, if an honest participant believes that a protocol  $\pi$  is insecure due to the prior information acquired by other parties, then no other protocol that implements the same functionality can be secure.

Note that the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  must be independent of the input distribution  $\mathfrak{D}$ , otherwise the ideal world attacker has more prior information than the real world attacker. Namely, if the attack  $\mathcal{A}^\circ$  in the ideal world uses the whole description of  $\mathfrak{D}$ , then the dependence leaks information about inputs and we lose privacy guarantees. Similarly, one should assume that  $\mathcal{A}^\circ$  corresponding to  $\mathcal{A}$  corrupts the same set of participants, otherwise the attack strategies  $\mathcal{A}$  and  $\mathcal{A}^\circ$  are incomparable. At the same time, the dependence on  $\mathcal{B}(\cdot)$  does not introduce any side-effects, since the predicate  $\mathcal{B}(\cdot)$  is assumed to be known to all attackers. In fact, an attacker is assumed to behave optimally w.r.t.  $\mathcal{B}(\cdot)$ .

More generally, let the set of relevant security objectives  $\mathcal{B}(\cdot)$  be denoted as  $\mathfrak{B}$ . Then a real and an ideal world model are  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t. the predicate set  $\mathfrak{B}$  if for any  $t_{\text{re}}$ -time  $\mathcal{A}$  and  $\mathcal{B}(\cdot) \in \mathfrak{B}$ , there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathfrak{D}$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (5.6)$$

Alternatively, we can view  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -closeness w.r.t.  $\mathfrak{B}$  as a reduction schema<sup>1</sup>

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon]{\text{MPC-SEC}} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \quad (5.7)$$

that can be applied to simplify analysis of security games, where the predicate  $\mathcal{B}(\cdot) \in \mathfrak{B}$  is computed interactively after the completion of the protocol  $\pi$ ;  $t_{\text{re}}$  limits the online complexity of an attack  $\mathcal{A}$  and the inequality (5.6) limits the discrepancy between  $\mathcal{G}_{\text{real}}^{\mathcal{A}}$  and  $\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ}$ . See Sections 7.2-7.5 for further details.

Finally, note that the exact dependencies  $t_{\text{id}} = t_{\text{id}}(t_{\text{re}})$  and  $\varepsilon = \varepsilon(t_{\text{re}})$  are important, as they determine the correspondence between the ideal and the real world time-success profiles  $\varepsilon_0(t)$  and  $\varepsilon_1(t)$ . Indeed, let  $t_\pi$  be the total running time of the protocol  $\pi$  and  $\varepsilon(t_{\text{re}})$  the right hand side of (5.6). Then any ideal world attack can be carried out in the real world with a constant  $t_\pi$ -time overhead. Similarly, the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  assures that any  $t_{\text{re}}$ -time real world attack has  $t_{\text{id}}$ -time ideal world counterpart. Consequently, we get

$$\varepsilon_0(t_{\text{re}} - t_\pi) \leq \varepsilon_1(t_{\text{re}}) \leq \varepsilon_0(t_{\text{id}}(t_{\text{re}})) + \varepsilon(t_{\text{re}}) . \quad (5.8)$$

In short, the time-success profiles are comparable  $\varepsilon_0(t) \approx \varepsilon_1(t)$  only if  $\varepsilon(t_{\text{re}})$  is small and time bounds  $t_{\text{re}}$  and  $t_{\text{id}}$  are comparable. Note that the inequality (5.8)

<sup>1</sup>The name of the reduction schema MPC-SEC emphasises the fact that the reduction schemata can be applied as the protocol  $\pi$  is secure in the multi-party sense.

is more precise if we assume that the computational effort done by the trusted third party  $\mathcal{T}$  is free for the ideal world adversary  $\mathcal{A}^\circ$ .

**The choice of relevant predicates.** The most conservative alternative is to include all predicates  $\mathcal{B} : \{0, 1\}^{n+1} \rightarrow \{0, 1\}$  into the set  $\mathfrak{B}$ . Such a choice implicitly states that the real world and the ideal world implementations lead to comparable outputs even if the computational process following the protocol is unbounded. We can talk about *statistical*  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -closeness of the real world and the ideal world if  $\mathfrak{B}$  consists of all possible predicates. Alternatively, we can assume that the adversary is computationally unbounded. Then bounds on the online complexities  $t_{\text{re}}$  and  $t_{\text{id}}$  become irrelevant and we can talk about  $\varepsilon$ -closeness of the real and the ideal world. Both notions of statistical closeness lead to externally consistent security models. However, the nature of these security guarantees is completely different, see Section 6.2 for further discussion.

Statistical closeness is unachievable for most cryptographic tasks, since many cryptographic primitives, like (asymmetric) encryption, hash functions and digital signatures, are secure only due to our limited computational abilities. Also, one often needs security guarantees only for limited time intervals. For example, in many countries secret governmental information is usually declassified after 25–75 years. Therefore, it makes sense to consider the set of time-bounded predicates  $\mathfrak{B}$  or add even more severe restrictions. In particular, we can talk about *computational*  $(t_{\text{re}}, t_{\text{pr}}, t_{\text{id}}, \varepsilon)$ -closeness between the real and the ideal world when the set  $\mathfrak{B}$  consists of all  $t_{\text{pr}}$ -time predicates. The latter gives security guarantees for all scenarios, where the total time needed to post-process the protocol outputs and to evaluate the success criterion is less than  $t_{\text{pr}}$ .

**Comparison with the standard approach.** As said before, our approach is slightly different from the mainstream formalisation [Can00a, Gol04]. In the classical formalisation, the correspondence  $\mathcal{A} \mapsto \mathcal{A}^\circ$  between real and ideal adversaries must be universal for all predicates  $\mathcal{B}(\cdot) \in \mathfrak{B}$  and input distributions  $\mathfrak{D}$ . In other words, given an adversary  $\mathcal{A}$ , we can find a corresponding universal adversary  $\mathcal{A}^\circ$  that works equally well for all possible security objectives  $\mathcal{B}(\cdot)$  and we get a seemingly stronger security guarantee. However, such an appearance is deceptive, since for each security goal  $\mathcal{B}(\cdot)$ , time bound  $t_{\text{id}}$ , level of prior information specified as  $\mathfrak{D}$  and risk tolerance, there exists an optimal generic attack  $\mathcal{A}_*^\circ$  that already depends on  $\mathcal{B}(\cdot)$ . Hence, even if the map  $\mathcal{A} \mapsto \mathcal{A}^\circ$  is independent of  $\mathcal{B}(\cdot)$ , we must still estimate the success in terms of  $\mathcal{A}_*^\circ$ :

$$\text{Adv}_{\mathcal{G}_{\text{real}}}(\mathcal{A}) \leq \text{Adv}_{\mathcal{G}_{\text{ideal}}}(\mathcal{A}^\circ) + \varepsilon \leq \text{Adv}_{\mathcal{G}_{\text{ideal}}}(\mathcal{A}_*^\circ) + \varepsilon . \quad (5.9)$$

Consequently, the independence from  $\mathcal{B}(\cdot)$  does not have any effect on time-success or risk profiles. Also, one cannot rationally eliminate the possibility that an attacker actually knows his or her attack target  $\mathcal{B}(\cdot)$  and can thus use both schemes to convert his or her attack to the universal attack  $\mathcal{A}^\circ$ .

In Section 7.3 we also show that both formalisations are quantitatively equivalent for static adversaries, i.e., the choice of formalism is just a matter of taste. From a practical viewpoint, there is no difference anyway, as standard security proofs provide a construction of  $\mathcal{A}^\circ$  that depends only on  $\mathcal{A}$ . Recall that reduction schemata are usually stated in terms of computational distance and consequently we get natural bounds for  $\text{cd}_{\mathbf{z}_{\text{obs}}}^{t_{\text{pr}}}(\mathcal{G}_{\text{re-atk}}^{\mathcal{A}}, \mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ})$ .

Another small but still important difference lies in the way we treat corruption. Note that corruption is not objectively observable, since it describes the

internal state of a participant, which may or may not influence the observable outcomes. In particular, a predicate  $\mathcal{B}(\cdot)$  cannot treat  $z_i$  differently depending on whether  $\mathcal{P}_i$  is corrupt or not, since such a dependence would lead to externally inconsistent security goals. In practice, one can detect corruption with high probability by forcing each participant  $\mathcal{P}_i$  to prove in zero-knowledge that he or she knows an input  $x_i$  and randomness  $\omega_i$  that would lead to the observed protocol transcript. However, the latter is already a new protocol.

**Various levels of security.** The correspondence between the real and the ideal world implementations is amazingly flexible. The description of the ideal world model, the tolerated adversarial behaviour and the set of relevant predicates  $\mathfrak{B}$  are the three fundamental factors that determine the corresponding security guarantees. Intelligent variation of these basic factors can produce myriads of relevant security models. However, four of them are more essential than the others, since they formalise somewhat canonical security objectives. The following four sections are dedicated to these fundamental models.

## 5.4 SECURITY IN SEMI-HONEST MODEL

Semi-honest model provides the weakest but still meaningful security guarantees. Briefly, semi-honest participants follow the protocol description, but quietly store all intermediate values and later try to deduce some extra information about the remaining inputs or outputs. Hence, semi-honest behaviour is always possible, unless we take extreme organisational measures to assure that internal computational results cannot be stored. Moreover, note that the following formal definition captures only the basic privacy requirements, as malicious behaviour is prohibited and the outputs cannot be tampered with.

**Definition 2.** A protocol  $\pi$  is  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -private w.r.t. the idealised functionality  $f_1, \dots, f_n$  and the adversarial structure  $\mathcal{A}$  in the semi-honest model if the corresponding real and ideal world models are  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -close. That is, for any  $t_{\text{re}}$ -time adversary  $\mathcal{A}$  and for any  $t_{\text{pr}}$ -time predicate  $\mathcal{B}(\cdot)$ , there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathfrak{D}$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (5.10)$$

Some important details in the definition are worth stressing. First, the time difference  $t_{\text{id}} - t_{\text{re}}$  quantifies the maximal computational gain that the adversary may achieve. Second, the bound  $t_{\text{pr}}$  determines the maximal time complexity of a post-processing phase that preserves security. Third, ideal world model is flexible enough to accommodate any general purpose security objective.

Non-cryptographers often complain that the formalism is overly restrictive in practice, since participants may tolerate bigger information leakage than is needed to compute the outputs. However, it is surprisingly simple to model such protocols by providing a proper description of an ideal world model. More precisely, assume that participants need to compute a functionality  $\hat{f}_1, \dots, \hat{f}_n$ , however, they are willing to leak  $f_1, \dots, f_n$  instead. This setting is meaningful only if the desired outputs can be computed from the leaked ones:

$$\hat{f}_i(x_1, \dots, x_n) = g_i(f_i(x_1, \dots, x_n)) . \quad (5.11)$$

Consequently, we can consider an ideal world model, where the participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  first use a trusted third party  $\mathcal{T}$  to get replies  $f_i(x_1, \dots, x_n)$  and after that locally compute  $\hat{y}_i = g_i(y_i)$ . That is, a non-corrupted party  $\mathcal{P}_i$  deletes  $y_i$  and outputs  $z_i = (x_i, \hat{y}_i)$ . The computational closeness between the real and the ideal world models implies that the semi-honest node  $\mathcal{P}_i$  cannot learn anything beyond  $y_i$  but at the same time the protocol  $\pi$  must correctly reproduce only the values  $\hat{y}_i$ . Therefore, we can easily analyse protocols where the outputs are strictly less informative than potentially leaked information. In the limiting case, one can reveal enough information, so that each participant can compute all protocol messages, i.e., the methodology is universally applicable.

As a concrete example, consider private evaluation of randomised approximation algorithms [FIM<sup>+</sup>06]. In such settings, the distribution of approximations  $\hat{f}_i(x_1, \dots, x_n)$  can be simulated knowing only  $f_i(x_1, \dots, x_n)$  and we can formalise the maximal leakage as  $f_1, \dots, f_n$ . Similarly, we can formalise security for the protocols that sometimes fail to produce the correct output. In fact, we can state the correctness requirement as a separate property.

**Definition 3.** *A failure probability  $\delta$  for a protocol  $\pi$  is the maximal probability that the obtained outputs  $\hat{y}_i$  differ from the intended outputs  $f_i(x_1, \dots, x_n)$  in the semi-honest model. A protocol  $\pi$  is called correct if there are no failures.*

One can often achieve significant gains in efficiency by allowing negligible failure probability. For example, many protocols for private predicate evaluation based on homomorphic encryption and oblivious polynomial evaluation have a small tunable failure probability. See the articles [Fis01, FNP04, BK04, KS05] for the most prominent examples and [LLM05, LL07] for the author’s results.

We emphasise that any efficiently computable functionality can be privately evaluated. Yao was the first theoretician to state these completeness theorems for the semi-honest model [Yao82, Yao86]. We refer to the manuscript [LP04] for the corresponding proofs and historical remarks. For the multi-party protocols, these completeness theorems were first proven by Goldreich, Micali and Wigderson for the computational setting [GMW87]. The information-theoretical setting where the adversary is unbounded was covered only a year later [BOGW88, CCD88]. See the articles [MR91b, Bea91b, Can00a, Gol04] for further historical references.

## 5.5 INPUT-PRIVACY IN MALICIOUS MODEL

Before going any further, note that the privacy in the semi-honest model captures two different security goals: input-privacy and output-privacy. Moreover, these goals are quite loosely coupled. As an example, consider a randomised protocol, where participants generate outputs locally without using inputs. Then the protocol preserves the privacy of these inputs even if participants broadcast all outputs. Similarly, the privacy of outputs is preserved when participants broadcast all inputs. For deterministic functionalities, output-privacy implies input-privacy, since additional knowledge about inputs leaks information about outputs. Also, it is evident that output-privacy and correctness guarantees are tightly coupled when the adversary can deviate from the protocol description, since the ability to influence outputs violates privacy constraints.

Consequently, input-privacy is one of the weakest security objectives that still makes sense in the *malicious model*, where a coalition of corrupted parties can arbitrarily deviate from the protocol description. Moreover, input-privacy itself is an economically justified design goal for some settings.

Consider a video on demand service as an illustrative example. Assume that a single subscription server delivers content decryption keys to all potential clients. The functionality and the threats are asymmetric for this setting. As there are thousands, if not millions, potential subscribers, it is impossible to guarantee semi-honest behaviour for all clients. However, we can still use organisational methods to assure that the server is semi-honest. Also, the service provider itself is economically motivated to ensure that an honest client gets the desired output. On the other hand, the service provider must limit potential information leakages against a coalition of malicious clients.

Input-privacy is formally defined in terms of interactive hypothesis testing, where a malicious adversary tries to verify hypotheses about protocol inputs. Although it is sufficient to bound the total running time, we separate online and offline attack complexities to achieve more precise classification. More formally, let the set  $\mathfrak{B}_{z_a}(t_{\text{pr}})$  consists of all  $t_{\text{pr}}$ -time predicates  $\mathcal{B}(z_1, \dots, z_n, z_a) = \mathcal{B}(z_a)$  that can be computed offline by the adversary in the following definition.

**Definition 4.** *A protocol is  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -input-private w.r.t. the idealised functionality  $f_1, \dots, f_n$  and the adversarial structure  $\mathcal{A}$  in the malicious model if the corresponding real and ideal world models are  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t.  $\mathfrak{B}_{z_a}(t_{\text{pr}})$ . That is, for any  $t_{\text{re}}$ -time adversary  $\mathcal{A}$  and for any  $t_{\text{pr}}$ -time predicate  $\mathcal{B}(\cdot)$  in the form  $\mathcal{B}(z_1, \dots, z_n, z_a) = \mathcal{B}(z_a)$ , there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathfrak{D}$*

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (5.12)$$

Input-privacy is strictly weaker than the complete security in the malicious model and thus input-private protocols are often more efficient. In particular, no non-trivial protocol can be secure in the malicious model and have only two rounds, see the discussions in [GK96b, KO04]. Nevertheless, many two-round protocols do achieve input-privacy. For example, standard security definitions of oblivious transfer [NP99a, AIR01] require input-privacy with respect to unbounded receivers and computationally bounded senders. The same security requirements are often posed for other two-round client-server protocols, where only one participant obtains the output. One of the articles included into the thesis [LL07] investigates a certain subclass of two round client-server protocols and provides an automatic compilation technique, which converts any private protocol in the semi-honest model to an input-private protocol in the malicious model. The transformation preserves round complexity and has a relatively small overhead for many practical protocols.

## 5.6 OUTPUT-CONSISTENCY IN MALICIOUS MODEL

For many applications, input-privacy is not enough as honest parties need also explicit correctness guarantees. In particular, malicious behaviour should be detectable and one should be able to issue verifiable complaints about cheating.

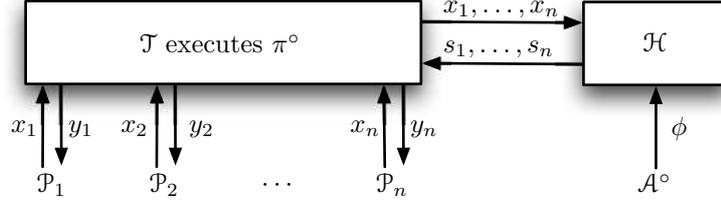


Figure 5.4: Ideal world model of output-consistent computations.

Now such correctness guarantees come in two flavours: honest parties may either detect malicious behaviour in general, or only the malicious changes in the outcomes. The second design goal can be viewed as output-consistency.

Observe that output-consistency is strictly weaker than complete security. Namely, if a malicious participant can alter his or her behaviour so that the outputs are changed only for few inputs  $\mathbf{x} \in \mathcal{X}$ , then a public complaint about such a change reveals that  $\mathbf{x} \in \mathcal{X}$ . The latter is not true for complete security, since the corresponding complaint only implies that the outputs could have been altered for some inputs. Therefore, output-consistency usually causes a significantly smaller overhead than complete security. Moreover, output-consistency is an economically justified design goal for scenarios, where trustworthy reputation is worth more than the potential gain achieved by cheating.

Informally, a protocol is output-consistent if a malicious adversary cannot tamper the outputs without causing honest parties to halt. More formally, an adversary has partial control over the ideal protocol  $\pi^\circ$ . A non-corruptible trusted party  $\mathcal{T}$  still computes the outcomes of a protocol, however, a *halting-machine*  $\mathcal{H}$  controls the delivery of outputs, see Fig. 5.4. That is, the halting-machine  $\mathcal{H}$  gets inputs  $x_1, \dots, x_n$  from  $\mathcal{T}$  and sends back zero-one verdicts  $s_1, \dots, s_n$ . Next, the trusted third party  $\mathcal{T}$  sends the output  $y_i$  to the participant  $\mathcal{P}_i$  only if  $s_i = 1$ . The communication between the halting-machine  $\mathcal{H}$  and the adversary  $\mathcal{A}^\circ$  is unidirectional—only the adversary  $\mathcal{A}^\circ$  can send messages to  $\mathcal{H}$ . Formally, it is sufficient to assume that  $\mathcal{H}$  is a universal computing device such that the adversary  $\mathcal{A}^\circ$  can start  $\mathcal{H}$  by sending the initial program code  $\phi$  and possibly later fine-tune  $\mathcal{H}$  by sending additional instructions.

**Definition 5.** A protocol is  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -output-consistent w.r.t. the idealised functionality  $f_1, \dots, f_n$  and the adversarial structure  $\mathcal{A}$  in the malicious model if the corresponding real and ideal world models are  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -close. That is, for any  $t_{\text{re}}$ -time adversary  $\mathcal{A}$  and for any  $t_{\text{pr}}$ -time predicate  $\mathcal{B}(\cdot)$ , there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathcal{D}$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (5.13)$$

First, observe that an output-consistent protocol must also be private in the semi-honest model and input-private in the malicious model. Second, the exact interaction pattern between  $\mathcal{A}^\circ$  and  $\mathcal{H}$  determines the maximal expressiveness of leaked predicates when an honest participant issues a rightful complaint. Therefore, it makes sense to characterise the underlying structure of these predicates. Sometimes one can show that  $\mathcal{H}$  always evaluates predicates that have a simple

structure, see our manuscript [LL06] for further discussion. Finally, the computational effort done by  $\mathcal{H}$  should be free for  $\mathcal{A}^\circ$  to make the correspondence between the ideal and the real world time-success profiles more tight.

The concept of output-consistency can be traced back to the classical works on secure multi-party computation [FY92, CFGN96, CO99], where authors consider detectability of malicious behaviour. However, none of them clearly formulates the ideal world model. The concept has also implicitly surfaced as a natural design property for adaptive oblivious transfer [NP99b], since it is almost trivial to achieve. Recently, Aumann and Lindell generalised these early notions and proposed three alternative security models [AL07]. However, their formulations are strictly weaker than output-consistency, as none of them guarantees even input-privacy nor the ability to detect altered outputs.

Output-consistency also alleviates the problem of false accusations. When a complaint is issued about malicious behaviour in an input-private protocol, one has to prove his or her innocence. Since tampering with outputs is detectable in an output-consistent protocol, the burden of proof lies on persons who raise complaints. In particular, malicious coalition cannot frame honest participants if one can use digital signatures to protect the authenticity of protocol messages. This property is specially useful in the client-server scenarios, where clients have different access rights and the server may rightfully fail for some queries, e.g. private inference control problem [WS04]. However, there is a slight difference between two- and multi-party protocols. In two-party protocols, an output failure exposes the malicious participant, whereas the latter is not a priori true in the multi-party setting. Hence, we can talk about a stronger form of consistency that also reveals some maliciously acting participants.

## 5.7 COMPLETE SECURITY IN MALICIOUS MODEL

The highest attainable security level in the malicious model is of course the computational closeness between the real world and ideal world without a halting-machine  $\mathcal{H}$ . In this case, malicious participants can cause only non-selective protocol failures and public complaints reveal nothing about inputs.

**Definition 6.** *A protocol is  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -secure w.r.t. the idealised functionality  $f_1, \dots, f_n$  and the adversarial structure  $\mathcal{A}$  in the malicious model if the corresponding real and ideal world models are  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -close. That is, for any  $t_{\text{re}}$ -time adversary  $\mathcal{A}$  and for any  $t_{\text{pr}}$ -time predicate  $\mathcal{B}(\cdot)$ , there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathfrak{D}$*

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (5.14)$$

A major drawback of secure protocols is the overhead needed to achieve security, since all participants have to prove that they obey the protocol without disclosing their inputs and random coins. The latter can be done with zero-knowledge proofs of correctness, using the famous GMW-compiler [GMW87], or with verifiable secret sharing [BOGW88, CCD88] in the information theoretical setting. See also the state of the art results [Pin03, KO04, LP07] for Yao's two-party computation and its extensions to multi-party case [BMR90].

It is well known that secure protocols can be practically intractable when the input size is large and functionalities have a large circuit complexity. Also, the

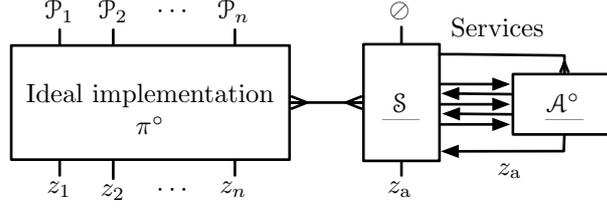


Figure 5.5: Canonical constructive correspondence between adversaries.

strength of organisational and physical methods is often enough to protect the computational processes and thus lower security levels are sufficient.

In particular, the difference between output-consistency and complete security can be marginal. Recall that issuing a complaint in output-consistent protocols reveals one bit of extra information. Consequently, such a leak may be a reasonable sacrifice for additional efficiency, if the protocol output itself reveals hundreds or thousands of bits. Moreover, adversaries must take a tremendous subjective risks to learn beyond 20 bits of new information. Namely, let  $\mathcal{D}$  be a subjective probability distribution that describes the adversaries prior knowledge about the inputs. Then the maximal entropy loss due to selective halting is  $H(s_1, \dots, s_n)$ , where  $s_1, \dots, s_n$  are halting decisions made by  $\mathcal{H}$  and the entropy is computed with respect to  $\mathcal{D}$ . Hence, the adversary must take a risk of being exposed with a subjective probability  $2^{-k}$  to learn  $k$  bits.

Another interesting aspect that is worth commenting is the issue of correctness guarantees raised by Micali and Rogaway [MR91a]. Namely, the computational closeness between the real and the ideal world models does not guarantee that honest participants obtain the correct values, rather they are  $t_{pr}$ -indistinguishable from the values obtained in the ideal world. We can naturally strengthen the correctness constraints, if we considers only the subclass of nice adversaries that append a flag `corrupted` to outputs of all corrupted participants. Namely, we must consider an additional set of predicates  $\mathcal{B}_{cor}$  that consists of all predicates that ignore the inputs  $z_i$  with the flag `corrupted`, i.e., require that the outputs of honest parties are statistically indistinguishable from the ideal world. Formally, this strengthening is externally inconsistent, since the corruption is not an externally observable phenomenon. Nevertheless, any hostile adversary can be converted to the corresponding nice adversary with minimal overhead so that the end result of a security game does not change. Hence, in principle such a strengthening can be incorporated into the model.

## 5.8 CANONICAL CONSTRUCTIVE CORRESPONDENCE

Note that all security definitions given above require a correspondence between the real and the ideal world adversaries. A canonical way to achieve such a correspondence is based on code wrappers that internally run a real world adversary and translate their actions into legitimate ideal world attacks, see Fig. 5.5. Briefly, a code wrapper  $\text{Sim}$  is guaranteed to succeed if it manages to simulate the protocol execution for an adversary  $\mathcal{A}$ . Thus, it is more common to use a term *simulator* instead. We emphasise that a simulator  $\text{Sim}$  has complete con-

trol over the real adversary  $\mathcal{A}$ . In particular, the simulator  $\text{Sim}$  provides random coins and timer services to  $\mathcal{A}$ . The simulator  $\text{Sim}$  can also restart or rewind  $\mathcal{A}$  to acquire extra knowledge or to bypass security checks in the protocol. All other participants see the algorithm pair  $(\text{Sim}, \mathcal{A})$  as an ideal world adversary  $\mathcal{A}^\circ$ , i.e.,  $\text{Sim}$  can corrupt ideal world participants and interact with the halt machine  $\mathcal{H}$ .

Most security proofs are based on black-box simulation, where the simulator  $\text{Sim}$  does not inspect nor manipulate the internal state and random coins of the adversary  $\mathcal{A}$ . Necessary information is gathered from the protocol messages exchanged between  $\text{Sim}$  and  $\mathcal{A}$ . The latter is not compulsory in white-box reductions, where  $\text{Sim}$  may act arbitrarily. Nevertheless, white-box reductions must be somehow constructive, or otherwise we are no closer to the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$ . Generally, white-box reductions use some sort of adversary dependent information to interact with the adversary  $\mathcal{A}$ .

Simulatability as a security criterion appeared only after the discovery of zero-knowledge proofs [GMR85] and early definitions of secure multi-party computations [Yao82, Yao86] were given in terms of maximal indistinguishability. For clarity, consider two-party computations with a corrupted node  $\mathcal{P}_1$  in the semi-honest model. Then an adversary  $\mathcal{A}$  should be unable to distinguish protocol runs corresponding to the inputs  $x_2$  and  $\hat{x}_2$  whenever  $f_1(x_1, x_2) = f_1(x_1, \hat{x}_2)$ . Now, if the adversary  $\mathcal{A}$  is good in predicting a function  $g(x_1, x_2)$ , then  $g$  must be constant over the class of inputs

$$\mathcal{M}_{x_1, y} = \{(x_1, x_2) : f_1(x_1, x_2) = y\} \quad (5.15)$$

or we get a contradiction with the basic properties of semantic security, see Section 3.2. Consequently,  $\mathcal{A}$  cannot predict anything that is not expressible in terms of  $x_1$  and  $y$ . However, the limitation does not apply for the functions  $g$  that can be computed from  $x_1$  and  $y$ . For example, assume that  $F$  is a permutation that is hard to invert and consider a protocol, where the participant  $\mathcal{P}_2$  sends a message  $e = F^{-1}(x_2)$  to  $\mathcal{P}_1$  who outputs  $y_1 = F(e) = x_2$ . Then the protocol runs corresponding to the same output  $y_1$  are indistinguishable by the construction. However, the participant  $\mathcal{P}_1$  learns  $F^{-1}(x_2)$ , which is difficult to obtain from the output  $x_2$  alone. Therefore, the approach based on the maximal indistinguishability is unsuitable, since it does not provide adequate security guarantees for all target functions  $g(x_1, x_2)$ .

Simulating the protocol execution in the ideal world can be quite challenging. Fortunately, the semi-honest model has two important properties that make the task easier. First, the adversary  $\mathcal{A}$  cannot change the inputs of corrupted parties. Second, the adversary  $\mathcal{A}$  must follow the protocol and it is sufficient if the simulator  $\text{Sim}$  can simulate all messages sent to corrupted parties. Let  $\mathcal{C}$  be the set of corrupted parties and  $\bar{\mathcal{C}}$  be its complement. Then simulating a static adversary  $\mathcal{A}$  is particularly straightforward. The simulator  $\text{Sim}$  first corrupts nodes that belong to  $\mathcal{C}$ , submits their inputs  $(x_i)_{i \in \mathcal{C}}$  to the trusted third party  $\mathcal{T}$  and uses the received values  $(f_i(\mathbf{x}))_{i \in \mathcal{C}}$  to simulate all messages sent to  $\mathcal{C}$  in the real execution. Adaptive and mobile corruption makes the task more difficult, since the simulator  $\text{Sim}$  must decide when to submit inputs to  $\mathcal{T}$  and assure that the adversary's view remains consistent after a new corruption instruction.

Providing a good simulation in the malicious model is more demanding, as a simulator  $\text{Sim}$  cannot automatically submit inputs  $(x_i)_{i \in \mathcal{C}}$ . Namely, a malicious adversary  $\mathcal{A}$  can always replace the initial input  $x_i$  of a corrupted node  $\mathcal{P}_i$  by

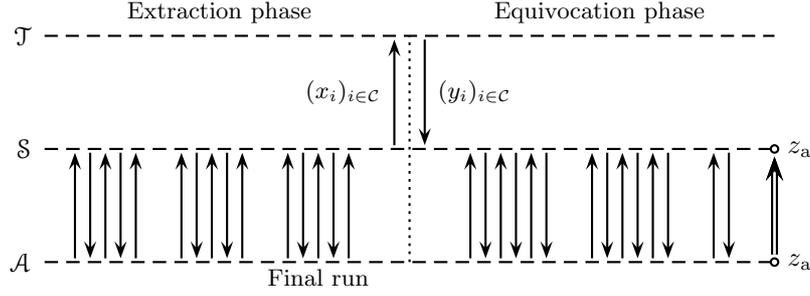


Figure 5.6: Canonical structure of a simulation procedure.

a new value  $\hat{x}_i$  and thus  $\text{Sim}$  must reconstruct the actual inputs  $(\hat{x}_i)_{i \in \mathcal{C}}$ . Consequently, the simulation must follow a specific structure depicted in Fig. 5.6. Briefly, the simulator must first extract all inputs of corrupted parties and then, given corresponding outputs, continue the simulation appropriately.

**Extractability.** If we want to achieve a black-box simulation, the simulator must extract inputs from the messages sent by corrupted participants. In particular, one should be able to extract the inputs of a semi-honest coalition. Such an extraction property seems to contradict input-privacy. However, such impression is misleading, as the powers of a simulator and adversary are always asymmetric. In the multi-party setting, the asymmetry is achieved by the clever choice of the adversarial structure  $\mathcal{A}$ . Namely, the adversarial structure  $\mathcal{A}$  is often chosen so that a complement  $\bar{\mathcal{C}}$  of a plausible coalition  $\mathcal{C} \in \mathcal{A}$  can never be corrupted. Thus, the simulator can form a malicious coalition  $\bar{\mathcal{C}}$  that is large enough to extract the inputs  $(x_i)_{i \in \mathcal{C}}$  from the received protocol messages, whereas the coalition  $\bar{\mathcal{C}}$  is impossible in the real protocol.

The same trick does not work for protocols with honest minority, where an adversary can corrupt as many or even more parties than a simulator. Corresponding security proofs use another important asymmetry. Namely, the simulator  $\text{Sim}$  can always rewind  $\mathcal{A}$  to previous states and then change messages sent to  $\mathcal{A}$ . Thus, the simulator  $\text{Sim}$  can restore  $\hat{x}_i$  by observing many correlated protocol transcripts, whereas the latter is impossible for an adversary  $\mathcal{A}$ .

**Equivocability.** Observe that a protocol is input-private only if a protocol binds all participants to their future outputs. For the sake of contradiction, assume that an adversary  $\mathcal{A}$  can legitimately change the inputs after the extraction of inputs  $(\hat{x}_i)_{i \in \mathcal{C}}$ . Namely,  $\mathcal{A}$  can efficiently find yet another set of inputs  $(\tilde{x}_i)_{i \in \mathcal{C}}$  and random coins  $(\omega_i)_{i \in \mathcal{C}}$  such that messages exchanged so far are consistent with the protocol. Then the adversary  $\mathcal{A}$  can clearly learn  $(\tilde{y}_i)_{i \in \mathcal{C}}$  instead of  $(\hat{y}_i)_{i \in \mathcal{C}}$  obtained by the simulator. Consequently, the new set of outputs  $(\tilde{y}_i)_{i \in \mathcal{C}}$  must be efficiently computable from the old ones  $(\hat{y}_i)_{i \in \mathcal{C}}$ . For most protocols, this implies that  $(\tilde{y}_i)_{i \in \mathcal{C}} \equiv (\hat{y}_i)_{i \in \mathcal{C}}$  for all legitimate input changes  $(\tilde{x}_i)_{i \in \mathcal{C}}$ .

For many protocols, such an implicit or explicit commitment to the outputs creates another problem, since the simulator  $\text{Sim}$  itself does not know  $(\hat{y}_i)_{i \in \mathcal{C}}$  in the extraction phase. As a result,  $\text{Sim}$  must be able to legitimately retrofit the outputs  $(\hat{y}_i)_{i \in \mathcal{C}}$ . More precisely, the simulator  $\text{Sim}$  must produce a continuation for the protocol run that leads to  $(\hat{y}_i)_{i \in \mathcal{C}}$  if  $\mathcal{A}$  behaves honestly after the extraction. Hence, we have a similar asymmetry as before. For malicious parties, the pro-

protocol must be binding, and for the simulator, the protocol must be equivocal. Again, a clever choice of the adversarial structure  $\mathcal{A}$  can create the desired asymmetry of powers for multi-party protocols. For two-party protocols, we must use rewinding to bypass consistency checks that force  $(\tilde{y}_i)_{i \in \mathcal{C}} \equiv (\hat{y}_i)_{i \in \mathcal{C}}$ . Moreover, it is often impossible to rewind the entire extraction phase to retrofit the desired outputs, as the latter may change the extractable inputs  $(\hat{x}_i)_{i \in \mathcal{C}}$ .

**Trusted setup.** Many protocols use a special setup procedure, where a trusted dealer either broadcasts a single message or provides a trustworthy public key infrastructure. Note that the trusted dealer exists only in the real world model and thus a simulator can corrupt the trusted dealer in the ideal world to achieve extractability and equivocability. Hence, we can construct non-rewinding simulators also for two-party protocols where rewinding is otherwise unavoidable. Since the rewinding techniques do not scale well for parallel compositions, the trusted setup procedures are the only known way to achieve composability in the two-party setting, see Section 7.7 for more detailed discussion.

To summarise, the canonical constructive correspondence is achievable only by explicit embedding of trapdoor mechanisms that violate both input-privacy and output-consistency. The latter causes a tension between the security and simulation, which is always resolved by introducing the asymmetry of powers between the simulator and the adversary. In less formal terms, no protocol can be “unconditionally” secure in all settings.

## 6 ALTERNATIVE SECURITY MODELS

Exact quantification of security properties is only one of many paradigms used in cryptography. Therefore, it is important to know the connections between exact security and its alternatives. We start from asymptotic security, since it is probably the most popular and widely studied alternative approach. Note that the initial simplicity of asymptotic security models is highly deceptive. In fact, even the notion of polynomial security varies in cryptographic literature. Therefore, we first formalise the limiting process that is needed to define asymptotic security and then explore common formalisations. As a result, we obtain the description of eight asymptotic security models and a corresponding hierarchy that characterises the qualitative relations between these models.

Subjective security models are another common alternative to exact security. The main difference between objective and subjective security models lies in the choice of possible adversaries. Objective security models can only restrict available computing resources, whereas subjective security models can place additional constraints on the set of potential adversaries. For example, all security definitions that consider only uniform adversaries are actually subjective, since a potential attacker is not allowed to choose algorithms that are optimal for concrete security parameters. We remark here that subjective security models are quite popular in cryptographic literature. For example, the random oracle, the ideal cipher and the generic group model are all subjective security models, since the corresponding security guarantees are meaningful in practice only if we are willing to make certain subjective assumptions about the set of possible adversarial algorithms. Finally, subjective security notions are also appropriate when we consider protocols with trusted setup, see Section 6.4.

Subjective and objective security models share many technical similarities but there are also important conceptual differences. Most importantly, it is impossible to formalise certain natural security properties without using the notion of subjective security. In Section 6.4, we discuss these limitations more thoroughly and explain why objective approach is destined to fail in some settings. In Section 6.5, we rigorously formalise the notion of subjective security that allows us to escape these limitations. We also show that the corresponding security proofs are technically very close to standard proofs. Namely, a standard security proofs is also valid in the subjective setting if it is strictly constructive. Nevertheless, it might be challenging to find strictly constructive proofs, especially since the proof methodology presented in Chapter 4 is actually non-constructive. Fortunately, most of these non-constructive proof steps can be eliminated. We describe the corresponding semi-automatic techniques in Section 6.7. As a result, we can still describe proofs by using game trees and reduction schemata, and thus hide tedious technical details.

### 6.1 SCALABILITY AND POLYNOMIAL SECURITY

It is easy to see that all conditional security proofs just reduce the amount of unjustified trust by isolating few essential simplistic assumptions that guarantee the security of the entire construction. However, all instantiations of these basic

primitives are currently heuristic and the corresponding security guarantees are essentially unknown. Although there are some numerical security estimates for these basic primitives, they are based on empirical experiments or on heuristic claims, and thus likely to change over time. Moreover, the amount of available computing power is also likely to increase in time. Therefore, it is not wise to design protocols so that the underlying primitives cannot be replaced.

Fortunately, cryptographic constructions and protocols are seldomly tied to specific building blocks, such as the 128-bit MD5 hash function, and can thus be “regularly” updated to take advantage of more powerful computing devices and ensure the proper safety margin against plausible attacks. As conditional security proofs abstract away all unnecessary implementation details, we can view a cryptographic construction as a parametrised template, i.e., we can increase the security level by using stronger but slower basic primitives.

Let  $\mathbb{k}$  be a *security parameter* that indexes both a sequence of basic primitives and the construction in question. For clarity, assume that security can be characterised by the advantage

$$\varepsilon(\mathbb{k}) = \max_{\mathcal{A} \in \mathfrak{A}(\mathbb{k})} \text{Adv}_{\mathcal{G}_{\mathbb{k}}}(\mathcal{A}) \quad , \quad (6.1)$$

where  $\mathcal{G}_{\mathbb{k}}$  captures the attack against the construction  $\pi_{\mathbb{k}}$  and  $\mathfrak{A}(\mathbb{k})$  is the set of all  $t(\mathbb{k})$ -time adversaries. W.l.o.g. we can assume that  $t(\mathbb{k})$  is an increasing function and  $\varepsilon(\mathbb{k})$  is a decreasing function. Let us also define a *relative security margin*

$$\delta(\mathbb{k}) = \frac{t(\mathbb{k})}{\tau(\mathbb{k})} \quad , \quad (6.2)$$

where  $\tau(\mathbb{k})$  denotes the overall running time of  $\pi_{\mathbb{k}}$ . Note that essential algorithmic and cryptographic properties of the *construction family*  $(\pi_{\mathbb{k}})$  are determined by the functions  $\tau(\cdot)$ ,  $t(\cdot)$ ,  $\varepsilon(\cdot)$ ,  $\delta(\cdot)$ . For example, one can determine appropriate security level  $\mathbb{k}$ . To achieve  $(t_0, \varepsilon_0)$ -security, one has to choose minimal  $\mathbb{k}_0$  such that  $\varepsilon(\mathbb{k}_0) \leq \varepsilon_0$  and  $t(\mathbb{k}_0) \leq t_0$  provided that running time  $\tau(\mathbb{k}_0)$  is still feasible in practice. To make similar economical decisions, one must consider rescalings that characterise true costs and gains. Since the underlying principle of analysis remains the same, we consider only the algorithmic aspects.

The choice of the security parameter is quite arbitrary, usually it describes important implementation details, such as the bit length of the RSA modulus. From a theoretical viewpoint it is advantageous to identify  $\mathbb{k}$  with the maximal running time  $\tau(\mathbb{k})$  that is needed to complete the protocol  $\pi_{\mathbb{k}}$ . Then we must consider only the properties of  $\varepsilon(\cdot)$  and  $\delta(\cdot)$ . For obvious reasons, a decreasing safety margin  $\delta(\cdot)$  implies that the construction becomes more vulnerable in the future. More generally, a construction is sustainable in the long run only if the function  $\delta(\mathbb{k})$  is increasing and  $\varepsilon(\mathbb{k})$  is decreasing fast enough. Hence, it is tempting to analyse only the limiting behaviour of  $\delta(\cdot)$  and  $\varepsilon(\cdot)$ .

When one considers asymptotic behaviour, polynomial-time algorithms are special. Mostly, because the class of polynomially bounded functions  $\mathbf{poly}(\mathbb{k})$  is the first nontrivial class of functions that contains the identity function and is closed under the multiplication and superposition. The latter is sufficient to prove that the notion of polynomial-time algorithms is machine independent and the corresponding set of algorithms is closed under recursive composition. Mainly because of these reasons, Cobham and Edmonds postulated already in

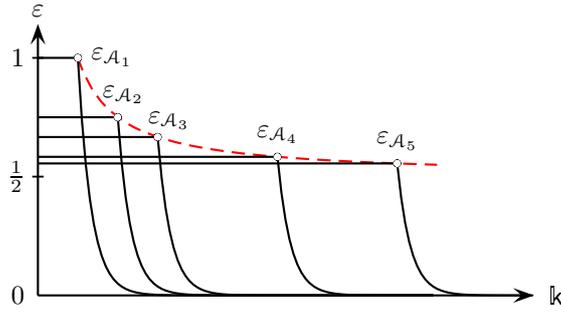


Figure 6.1: A construction may be completely insecure, although for each polynomial time bound  $t(k)$  the advantage  $\varepsilon_{\mathcal{A}_i}(k) \in \mathbf{negl}(k)$  for any algorithm  $\mathcal{A}_i$ .

the early 1960s that the set of feasible and polynomial algorithms must coincide [Cob64, Edm65]. This description of tractability was quickly adopted by other theoreticians and it led to the birth of the complexity theory.

Cryptographers reached an analogous consensus and postulated that a construction family  $(\pi_k)$  is *asymptotically secure* if for all polynomial safety bounds  $\delta(k) \in \mathbf{poly}(k)$  the corresponding advantage decreases faster than a reciprocal of any polynomial, i.e.,  $\varepsilon(k) \in \mathbf{negl}(k)$ . More precisely, we can talk about uniform and non-uniform polynomial security model. The essential difference between the models is in the definition of the set  $\mathfrak{A}(k)$ . In the *non-uniform model*, the set  $\mathfrak{A}(k)$  consists of all  $t(k)$ -time algorithms and thus represents objective security. The *uniform model* is a subjective security model, since the model assumes that an adversary uses a single self-tunable algorithm  $\mathcal{A}$  for all attacks. Formally, the set  $\mathfrak{A}(k)$  consists of a single  $t(k)$ -time algorithm  $\mathcal{A}_k$  that is obtained from  $\mathcal{A}$  by feeding  $k$  as the first argument. As the universal algorithm  $\mathcal{A}$  is unknown, the advantage  $\varepsilon_{\mathcal{A}}(k)$  must be asymptotically negligible for all algorithms  $\mathcal{A}$  and polynomial time bounds  $t(k) \in \mathbf{poly}(k)$ .

The uniform security model is completely natural under a super-strong artificial intelligence assumption. If we *believe* that any intelligent behaviour can be outperformed by an artificial intelligence—an algorithm  $\mathcal{A}$ , then we may indeed set  $\mathfrak{A}(k) = \{\mathcal{A}_k\}$  and study the limitations of intelligent reasoning. For more conservative readers, it is evident that uniform security model provides questionable security guarantees. Namely, security in the uniform model does not exclude the possibility that a potential adversary may achieve constant advantage by cleverly choosing algorithms for each instance, as depicted in Fig. 6.1. From this perspective, the naming convention is awfully misleading, as the uniform model assures only pointwise convergence of security guarantees, whereas the non-uniform model assures fast uniform convergence over all algorithms.

On a more practical note, observe that advances in computing technology show rather interesting trends. Although available computing power has grown in an exponential pace, the approximate ratio between ordinary and maximal available computing power has always been more or less constant with occasional fluctuations. Fig. 6.2 illustrates these trends by comparing the computational capabilities of widely accessible computing devices and the fastest supercomputers throughout the entire history of computing. In particular, note that the available adversarial power  $t$  has always been in the range

$$10 \cdot \tau \leq t \leq 100,000 \cdot \tau, \quad (6.3)$$

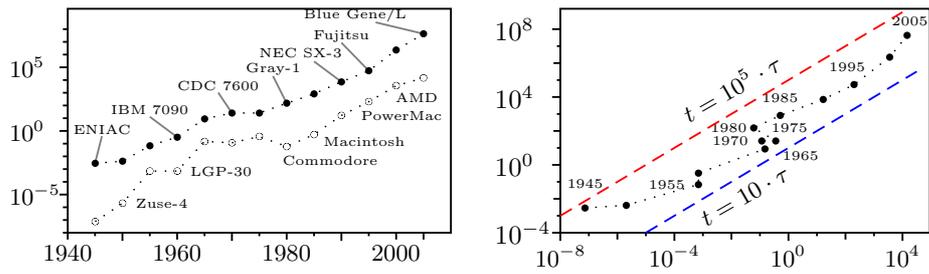


Figure 6.2: Comparison of conventional and supercomputers on the left and the corresponding dependence between honest and adversarial power on the right. Computing power measured in MIPS and the data is taken from [Mor98].

where  $\tau$  is the power of an ordinary computing device. Of course, the capabilities of the fastest supercomputer do not adequately characterise the total computing power available in the world. On the other hand, this measure clearly indicates how much monetary resources organisations are willing to spend on extremely important computational tasks and thus the estimate is adequate for practical purposes. Consequently, one can argue that the polynomial security model is overly conservative in the context of short-term security. In particular, non-existence proofs in the asymptotic security models are not convincing, unless they hold for all superlinear time bounds  $t(k) \in \omega(k)$ . The latter was the main reason why we established such strong non-existence proofs in [LN06].

Note that the ratio between ordinary and adversarial power is bounded even for longer time periods provided that the growth rate is not super-exponential. Of course, we cannot make irrefutable conclusions, since the empirical evidence is rather limited and not collected for cryptographic purposes. Still, one should be extremely careful in interpreting results that are obtained in the asymptotic setting. Both positive and negative results obtained in the asymptotic security model are usually inconclusive, unless they have explicitly stated and practical time bounds. For example, the first proposal for the public key cryptosystem [Mer78]—so called Merkle’s puzzle system—has only a quadratic relative safety margin and thus it is insecure in polynomial security model. However, in practice an ordinary computer can do  $10^{11}$  operations in minutes, whereas  $10^{22}$  operations takes roughly 1 year on Blue Gene/L. In other words, a quadratic security margin is already sufficient for short time security and the gap is growing with every year, since the computing power becomes cheaper and cheaper.

On the other hand, many constructions that are asymptotically secure under reasonable assumptions, such as Ajtai-Dwork cryptosystem [AD97], are insecure for all practical parameter sizes [NS98]. Going even further, note that a generic encoding of an **NP**-problem as a CNF-SAT formula leads to quadratic blow-up and thus a quadratic safety margin might be preserved even if CNF-SAT problems can be solved in linear time. As a result, even a constructive proof for the statement  $\mathbf{P} = \mathbf{NP}$  does not rule out the existence of practical cryptographic primitives. Similarly, the existence of asymptotically secure cryptography does not rule out the existence of efficient adversarial algorithms for all practical time bounds. To summarise, life is not so dull as Impagliazzo’s famous five world model [Imp95] predicts: cryptography might be a booming business even in

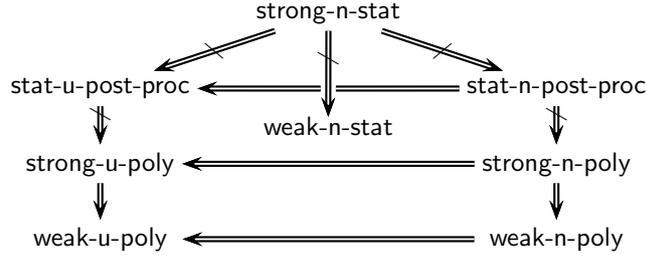


Figure 6.3: Relations between asymptotic security models. Hatched arrows denote implications with known separations, the others are believed to be strict. Symbols -u- and -n- denote uniform and non-uniform complexity.

Algorithmica, where the statement  $\mathbf{P} = \mathbf{NP}$  holds; whereas obtaining exact security bounds might be troublesome in Cryptomania, where asymptotically secure one-way trapdoor permutations are guaranteed to exist.

## 6.2 ASYMPTOTIC SECURITY FOR PROTOCOLS

Classifying protocols according to asymptotic properties is not as simple as it seems at first glance. As before, assume the maximal total running time of all honest participants  $\tau(\mathbb{k}) = \mathbb{k}$ . We emphasise here that the bound  $\tau$  must hold for any plausible attack pattern, i.e.,  $\tau$  quantifies the maximal workload that attacker can cause. Now regardless of the ideal world description, the security of a protocol  $\pi$  is characterised by a quadruple  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ . More precisely, the standard way to define the advantage<sup>1</sup> is the following

$$\varepsilon(\mathbb{k}) = \text{cd}_*^{t_{\text{pr}}(\mathbb{k})}(\mathcal{G}_{\text{re-atk}}^{\mathbb{k}}, \mathcal{G}_{\text{id-atk}}^{\mathbb{k}}) , \quad (6.4)$$

where games  $\mathcal{G}_{\text{re-atk}}^{\mathbb{k}}$  and  $\mathcal{G}_{\text{id-atk}}^{\mathbb{k}}$  capture the effects of active attacks. The game  $\mathcal{G}_{\text{re-atk}}^{\mathbb{k}}$  models the attack of  $t_{\text{re}}(\mathbb{k})$ -time real world adversary  $\mathcal{A}$  against the protocol  $\pi_{\mathbb{k}}$ . Similarly,  $\mathcal{G}_{\text{id-atk}}^{\mathbb{k}}$  models the attack of  $t_{\text{id}}(\mathbb{k})$ -time adversary  $\mathcal{A}^\circ$  in the ideal world. Observe that time bounds  $t_{\text{re}}(\cdot)$ ,  $t_{\text{id}}(\cdot)$  and  $t_{\text{pr}}(\cdot)$  can be arbitrarily fixed and thus there are several ways to define the limiting process  $\mathbb{k} \rightarrow \infty$ .

**Strong and weak polynomial security model.** Recall that actual security goals are always specified by individual security games and the ideal world model is just a mind experiment that simplifies the reasoning for such games. Hence, the polynomial security model for protocols should be consistent with the definition given in the previous section. Indeed, the standard definition postulates that a protocol family  $(\pi_{\mathbb{k}})$  is *secure in a (strong) polynomial model* if for any  $t_{\text{re}}(\mathbb{k}), t_{\text{pr}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists  $t_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  such that  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . Hence, for any polynomial-time security game the advantage in the real and the ideal world differ by a negligible amount. As the ideal world adversary is also polynomial, the corresponding advantage  $\varepsilon_{\mathcal{A}^\circ}(\mathbb{k})$  cannot be avoided for any cor-

<sup>1</sup>In the previous chapter, we used a slightly more liberal notion, where the construction of  $\mathcal{A}^\circ$  could depend on the target relation  $\mathcal{B}$ . But Section 7.3 shows that the latter is equivalent to the classical definition if we discard polynomial factors.

rect implementation of the desired functionality. Thus, a properly scaled advantage that omits the trivial success probability is indeed negligible.

However, soon after the wide acceptance of polynomial security model, cryptographers failed to prove the security of many interesting protocols. Remarkably, the first zero-knowledge proof [GMR85] is not secure in the strict polynomial model. The corresponding journal article [GMR89] had to explicitly relax security requirements and thus implicitly defined a weaker security model.

The weakened model bounds only the expected running time  $\hat{t}_{\text{id}}$  of the ideal adversary  $\mathcal{A}^\circ$ , where the average is taken over its coin tosses. The corresponding security definitions for zero-knowledge [GMR89, GMW91] and proofs of knowledge [BG92] require that for any  $t_{\text{re}}(\mathbb{k}), t_{\text{pr}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists a bound on average running time  $\hat{t}_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  such that  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . Although this change is seemingly marginal, it leads to several unexpected consequences, see [BL04, KL05]. The main drawback lies in the fact that a sequential composition theorem for zero-knowledge proofs [GO94] holds only in the strong polynomial model. Moreover, Barak and Lindell showed that there can be no constant round black-box zero-knowledge proofs for non-trivial problems in the strong polynomial model [BL04] and thus achieving constant round complexity and composability simultaneously is difficult. As a small consolation, they also showed that under reasonable cryptographic assumptions, there exists a constant round zero-knowledge protocol for any **NP** language with a white-box reduction but the corresponding security guarantees were quite loose.

Such a sad state of affairs has led to several technical refinements [KL05, Gol07] that avoid the unpleasant dilemma. Briefly, the refinements show that if the distribution of running time satisfies certain specific constraints, then the composition theorems hold for the model, where only the expected running times  $\hat{t}_{\text{re}}(\mathbb{k})$  and  $\hat{t}_{\text{id}}(\mathbb{k})$  are known to be polynomial. More importantly, almost all zero-knowledge proofs, including classical constant round proofs [GMW91, GK96a, FS89] for all **NP** languages, satisfy these constraints. Therefore, one can indeed avoid the choice between round efficiency and composability.

However, if one looks deeper into the issue, he or she soon discovers that the expected running time is just a camouflage for hiding unpleasant facts. Namely, certain properties of proofs of knowledge make the strict polynomial time black-box zero-knowledge impossible. In a nutshell, if a cheating prover succeeds with a probability  $\varepsilon_{\text{pok}}$ , then the corresponding knowledge-extraction algorithm must work in time  $\Theta(\varepsilon_{\text{pok}}^{-1})$  to succeed with overwhelming probability. Such a lower bound on the running time has a natural explanation: a black-box knowledge-extractor can succeed only if it gets a valid proof of knowledge and the latter requires on average  $\Omega(\varepsilon_{\text{pok}}^{-1})$  repetitions. A corresponding zero-knowledge proof that uses a proof of knowledge as a sub-protocol runs in expected polynomial time only because with probability  $1 - \varepsilon_{\text{pok}}$  a cheating verifier fails in the proof of knowledge, and thus one has to run the slow knowledge-extraction algorithm with probability  $\varepsilon_{\text{pok}}$ . Consequently, the unknown probability  $\varepsilon_{\text{pok}}$  cancels out and we are left with the expected running time  $\hat{t}_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$ .

However, such a trick does not solve the underlying intrinsic difficulties with knowledge extraction, since “useful computations” are done only in long runs. Moreover, the corresponding security guarantees are machine dependent, provide only average-case security, and have thus questionable interpretation in practice [BL04, p. 785]. Therefore, we choose a more obvious and sincere ap-

proach to relax polynomial security model. We say that a protocol family  $(\pi_{\mathbb{k}})$  is *weakly secure in the polynomial model* if for any asymptotically non-negligible bound  $\varepsilon(\mathbb{k}) \in \Omega(\mathbb{k}^{-c})$  and for any  $t_{\text{re}}(\mathbb{k}), t_{\text{pr}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists an ideal world adversary with a strict running time  $t_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$ .

The corresponding security model is not novel at all. In fact, the original definition of proofs of knowledge [FFS87] was indeed given in the weak polynomial security model. Later the same concept resurfaced under the name of  $\varepsilon$ -knowledge [DNS04]. It is straightforward to show that the classical definition for proof of knowledge [BG92] also satisfies the original definition [FFS87] and thus all troublesome zero-knowledge proofs are secure in the weak model. More importantly, the security guarantees have clear interpretations and all composition theorems that hold in strong polynomial model also hold in the weak model. Hence, the author believes that the explicit definition of weak polynomial security model is better than various technical tweaks [KL05, Gol07].

Evidently, the strong polynomial security model is contained in the weak polynomial security model. Moreover, Barak and Lindell have proven the non-existence of constant round black-box zero-knowledge proofs for all languages  $\mathcal{L} \notin \mathbf{BPP}$  [BL04]. Thus, the aforementioned inclusion is strict for black-box reductions under a widely believed assumption  $\mathbf{NP} \neq \mathbf{BPP}$ . For white-box reductions, it is not known whether the inclusion is strict or not. At the same time, it is straightforward to prove that if a protocol is secure in a weak non-uniform polynomial security model, then for any  $t_{\text{re}}(\mathbb{k}), t_{\text{pr}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists an ideal world adversary with super-polynomial but sub-exponential running time  $t_{\text{id}}(\mathbb{k}) \in \mathbb{k}^{\omega(1)} \cap 2^{o(\mathbb{k})}$  such that the advantage is negligible  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ .

The proof is based on a simple diagonalisation argument. Fix bounds on the running times  $t_{\text{re}}(\mathbb{k}), t_{\text{pr}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  and consider the sequence of decreasing bounds  $\varepsilon_i(\mathbb{k}) = \mathbb{k}^{-i}$ . Then, by the definition, we also get a corresponding sequence of asymptotic polynomial time bounds  $t_{\text{id},i}(\mathbb{k}) \leq \mathbb{k}^{c_i}$  for large enough  $\mathbb{k}$ . Now, if we fix a sequence of switching points  $(\mathbb{k}_i)_{i=1}^{\infty}$  and use the  $i^{\text{th}}$  reduction in the range  $\mathbb{k} \in [\mathbb{k}_i, \mathbb{k}_{i+1})$ , then by the construction the resulting bound on the advantage  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . To complete the proof, we must choose switching points so that the running time is sub-exponential. If we take  $\mathbb{k}_i = 2^{c_i+1}$ , then  $t_{\text{id}} \in O(\mathbb{k}^{\log \mathbb{k}})$  but in principle, any super-polynomial bound is achievable.

Analogously, we can define the strong and the weak uniform polynomial security model. However, these notions make sense only if the protocol inputs are generated by a uniform polynomial algorithm. Otherwise, the adversary can use the inputs of corrupted participants as an external advice and we are still in the standard non-uniform setting. This additional requirement for inputs makes it difficult to interpret the resulting security guarantees. Hence, uniform security models are seldomly used. Nevertheless, it is the only plausible choice when the basic primitives are secure solely in the uniform model.

**Three alternatives for statistical security.** The first and the most obvious choice is to consider adversaries that are computationally bounded on the online phase of the attack, but afterwards have infinite time to analyse the obtained results. More formally, let us consider redefined advantage

$$\varepsilon(\mathbb{k}) = \text{sd}_{\star}(\mathcal{G}_{\text{re-atk}}^{\mathbb{k}}, \mathcal{G}_{\text{id-atk}}^{\mathbb{k}}) . \quad (6.5)$$

Then a protocol family  $(\pi_{\mathbb{k}})$  is *secure against polynomial online and unbounded post-processing attacks* if for any  $t_{\text{re}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists  $t_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$

such that  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . Similarly to the discussion above, we can relax the security conditions, i.e., a protocol family  $(\pi_{\mathbb{k}})$  is *weakly secure against polynomial online and unbounded post-processing attacks* if for any non-negligible error bound  $\varepsilon(\mathbb{k}) \in \Omega(\mathbb{k}^{-c})$  and for any  $t_{\text{re}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$  there exists a polynomial bound  $t_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k})$ . Since the adversary can store all received messages, two-party protocols that implement non-trivial functionality cannot be secure against unbounded post-processing attacks, see for example [CK89]. Nevertheless, these models are meaningful for certain two-party protocols that implement limited functionality. Various settings for statistical and perfect zero-knowledge (such as [BC86, Cha86, For87]) are most utilised examples but there are others, such as data authentication, see for example [BR93a].

Many multi-party protocols can also handle online attacks with unbounded time complexity. However, there are two ways of relaxing the online complexity. The most simplistic way is to postulate that the adversary has an indeed infinite computing power and thus the computational complexity is irrelevant. Under such assumptions it is natural to say that a protocol family  $(\pi_{\mathbb{k}})$  is *secure in the weak statistical security model* if for any time bound  $t_{\text{re}}(\mathbb{k})$  there exists a corresponding time bound  $t_{\text{id}}(\mathbb{k})$  that achieves  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . Although the definition limits the amount of extra information the adversary can gain, it does not limit the extra computational power that he or she can achieve by participating in the protocol. As a result, a protocol that is secure in the weak statistical model can still be insecure in the polynomial model. An excellent example of such a protocol is given in the seminal article [Can00a, p. 159].

Alternatively, we can require that the running times of the real and the ideal world adversaries are comparable for all functions  $t_{\text{re}}(\mathbb{k})$ . More formally, a protocol family  $(\pi_{\mathbb{k}})$  is *secure in the strong statistical security model* if for any function  $t_{\text{re}}(\mathbb{k})$  there exists a bound  $t_{\text{id}}(\mathbb{k}) \in \mathbf{poly}(\mathbb{k}) \cdot t_{\text{re}}(\mathbb{k})$  such that  $\varepsilon(\mathbb{k}) \in \mathbf{negl}(\mathbb{k})$ . This definition assures that the adversary cannot use the protocol to significantly speed up his or her computations. Thus, the strong statistical security model is a natural strengthening of all other asymptotic security models. The latter makes it the most dominant in the literature [Gol04, Can00a]. Also, all generic multi-party computation techniques [BOGW88, CCD88, GRR98] satisfy this definition. We emphasise that the strong statistical security model limits the discrepancy between the time-success profiles of the real and the ideal world. Intuitively, strong statistical security assures that a collapse of a computational basic primitive cannot cause abrupt jump in the time-success profile.

The weak statistical security model loses this correspondence between the time-success profiles of the real and the ideal world. As a result, the weak statistical security model has several weird and unexpected properties. Nevertheless, the definition is externally consistent. If one really assumes that the adversary has infinite computational resources, the gain in computational power is irrelevant and the honest parties must accept consequences of all possible attacks in the ideal world. In some practical applications, honest parties indeed care about specific properties that hold in the ideal world even for unbounded adversaries. For example, the concept of weak statistical security is often used in the context of oblivious transfer [AIR01, Lip05] and other two-party privacy-preserving operations, see for example [FNP04, FIPR05, KS05, LL07].

### 6.3 GENTLE INTRODUCTION TO SUBJECTIVE SECURITY

At first glance, the concept of subjective security seems a bit odd, not to say inappropriate, for a rational scientific discipline, such as cryptography. As scientists, we should seek impartial and well-justified descriptions of security threats and avoid any subjective and questionable assumptions on the adversarial behaviour. However, a closer inspection reveals that such models are quite common in mainstream literature. Even the celebrated uniform security model is strongly subjective, as there are no *objective* reasons to assume that an attacker will always use the same generic solution for all instances of a protocol.

The random oracle model is another widely used but often misunderstood security model that provides only subjective security guarantees. Bellare and Rogaway proposed the random oracle model [BR93b] as an idealised computational model to mimic hash functions. More formally, assume that a function  $h$  is chosen uniformly from the set  $\mathcal{H}_{\text{all}}$  where  $\mathcal{H}_{\text{all}} = \{h : \mathcal{M} \rightarrow \mathcal{T}\}$  consists of all functions from the set  $\mathcal{M}$  to the set  $\mathcal{T}$ . Now, let  $\mathcal{O}$  be an oracle that provides black-box access to the function  $h$ , i.e., given  $m \in \mathcal{M}$  the oracle replies  $h(m)$ . As a result, we have defined a computational model, where the oracle  $\mathcal{O}$  can be used as an idealised hash function from the set  $\mathcal{M}$  to the set  $\mathcal{T}$ .

Assume that a game  $\mathcal{G}$  describes the desired properties of a construction or a protocol. Then the definition of the random oracle model yields

$$\text{Adv}_{\mathcal{G}}^{\text{rom}}(\mathcal{A}) = \frac{1}{|\mathcal{H}_{\text{all}}|} \cdot \sum_{h \in \mathcal{H}_{\text{all}}} \Pr [\mathcal{G}^{\mathcal{A}, \mathcal{O}} = 1 | \mathcal{O}(\cdot) = h(\cdot)] \quad . \quad (6.6)$$

In practice, one must substitute  $\mathcal{H}_{\text{all}}$  with another function family  $\mathcal{H} \subseteq \mathcal{H}_{\text{all}}$  that has a more succinct description and can be efficiently evaluated. As a result, the advantage in the standard model is computed over a much smaller sample

$$\text{Adv}_{\mathcal{G}}^{\text{std}}(\mathcal{A}) = \frac{1}{|\mathcal{H}|} \cdot \sum_{h \in \mathcal{H}} \Pr [\mathcal{G}^{\mathcal{A}, \mathcal{O}} = 1 | \mathcal{O}(\cdot) = h(\cdot)] \quad . \quad (6.7)$$

A small sample size  $|\mathcal{H}| \ll |\mathcal{H}_{\text{all}}|$  by itself is not a problem. If a family  $\mathcal{H}$  is randomly sampled from  $\mathcal{H}_{\text{all}}$  then by the central limit theorem

$$\text{Adv}_{\mathcal{G}}^{\text{rom}}(\mathcal{A}) \approx \text{Adv}_{\mathcal{G}}^{\text{std}}(\mathcal{A}) \quad (6.8)$$

holds with overwhelming probability over the choice of  $\mathcal{H}$ . However, the function family  $\mathcal{H}$  must have a very short description and the latter introduces a possible bias into the estimate (6.8). Hence, a security proof in the random oracle model is convincing only under a *subjective* equivalence assumption that the approximation (6.8) still holds for the set of all *relevant* adversaries  $\mathfrak{A}$ .

Here, the concept of *relevant adversary*  $\mathcal{A} \in \mathfrak{A}$  needs further clarification. Although humans have admirable intellectual abilities, these powers are still limited. Therefore, humans are likely to choose a suboptimal adversarial strategies and one should not worry about optimal attacks that never materialise. The set  $\mathfrak{A}$  is meant to denote all algorithms that a mankind can devise. Of course, it is impossible to describe the set  $\mathfrak{A}$  ahead and thus we can *objectively* verify the claim (6.8) only after the attacks have taken place.

Nevertheless, one can try to describe a set of algorithms such that the subjective equivalence assumption is satisfied for this set. Assume that  $\mathcal{H}$  is a  $(t, \varepsilon)$ -pseudorandom function family, i.e., the security games

$$\begin{array}{c} \mathcal{Q}_0^{\mathcal{A}} \\ \left[ \begin{array}{l} h \leftarrow \mathcal{H} \\ \text{return } \mathcal{A}^{\odot} \end{array} \right. \end{array} \qquad \begin{array}{c} \mathcal{Q}_1^{\mathcal{A}} \\ \left[ \begin{array}{l} h \leftarrow \mathcal{H}_{\text{all}} \\ \text{return } \mathcal{A}^{\odot} \end{array} \right. \end{array}$$

are  $(t, \varepsilon)$ -indistinguishable. Now, consider a class of generic  $t$ -time attack algorithms  $\mathfrak{A}_{\text{bb}}$  that always evaluate  $h$  in a *black-box* manner. By definition

$$\text{Adv}_{\mathcal{G}}^{\text{std}}(\mathcal{A}) \leq \text{Adv}_{\mathcal{G}}^{\text{rom}}(\mathcal{A}) + \varepsilon, \quad (6.9)$$

for any  $\mathcal{A} \in \mathfrak{A}_{\text{bb}}$ , otherwise we can use the adversary  $\mathcal{G}^{\mathcal{A}}$  to distinguish between the games  $\mathcal{Q}_0$  and  $\mathcal{Q}_1$ . Thus, any attack that significantly violates the bound (6.9) must use the description of  $h$  in an essential way. As such an attack is specialised for  $h$ , it is likely not to succeed for other functions  $h^* \in \mathcal{H}$ . At the same time, a black-box adversary  $\mathcal{A} \in \mathfrak{A}_{\text{bb}}$  can still exploit the description of the family  $\mathcal{H}$  and thus the corresponding attacks can be quite specific. Hence, security in the random oracle model is a reasonable starting point whenever the hash function must exhibit pseudorandom properties in the construction.

The fact that the random oracle model is a subjective security model makes it quite different from objective security models. In particular, the validity of the subjective equivalence assumption (6.8) is not automatic and must be individually re-evaluated for each specific construction. Even a single decision is not universal, since different persons may reach different conclusions. As a result, a specific subjective equivalence assumption that is associated with a specific construction  $\pi$  can be falsified only by showing a concrete algorithm  $\mathcal{A}_0$  that for this concrete construction  $\pi$  achieves

$$\text{Adv}_{\mathcal{G}_{\pi}}^{\text{rom}}(\mathcal{A}_0) \ll \text{Adv}_{\mathcal{G}_{\pi}}^{\text{std}}(\mathcal{A}_0) . \quad (6.10)$$

Since one must make an individual subjective decision for each construction separately, the subjective equivalence assumption cannot be falsified by showing the existence of specially crafted protocols  $\pi_*$  that indeed satisfy (6.10). If we acknowledge this essential property of the model, the wave of critique [CGH04b, GK03, BBP04, MRH04, CGH04a] started by Canetti, Goldreich and Halevi falls apart. Essentially, all these results show that the subjective equivalence assumption (6.8) is not universal. But the latter is an expected result, otherwise the assumption (6.8) would be a valid mathematical claim.

The random oracle model is not the only widely used computational model that forces cryptographers to make subjective assumptions in order to interpret the results. All so-called black-box models starting from the ideal cipher model already proposed by Shannon [Sha49] and ending with the generic group and ring models [Nec94, Sho97, LR06] force cryptographers to make subjective equivalence assumptions similar to the approximation (6.8). Again, several extensions [Den02, Bla06a] of basic techniques [CGH04b, MRH04] indeed show that the corresponding equivalence assumption is not universal.

Finally, let us show that such a subjective equivalence assumption can be disputed in a more persuading manner. Assume that instead of a random oracle one uses standard iterative hash function family like SHA-1 and WHIRLPOOL. The

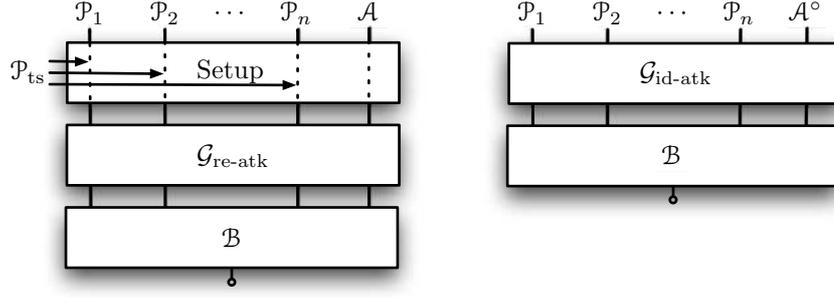


Figure 6.4: Real and ideal world models for a protocol with a trusted setup phase.

vanilla hash function  $f_{iv}^*$  without an appropriate padding is computed iteratively according to the Merkle-Damgård construction

$$h(x) = f_{iv}^*(x_1, \dots, x_k) = f(f_{iv}^*(x_1, \dots, x_{k-1}), x_k) , \quad (6.11)$$

where  $f : \{0, 1\}^m \times \{0, 1\}^n \rightarrow \{0, 1\}^m$  is a round function,  $iv$  is a  $m$ -bit initial value and  $x$  is parsed as a vector  $(x_1, \dots, x_k)$  of  $n$ -bit blocks. Consequently, we have a hash function family that is indexed by the initial value  $iv$ :

$$\mathcal{H}_f = \{f_{iv}^* : \{0, 1\}^{n^*} \rightarrow \{0, 1\}^m\}_{iv \in \{0, 1\}^m} . \quad (6.12)$$

Now given a value  $f_{iv}^*(x_1)$  and the black-box access to the round function  $f$ , it is straightforward to compute  $f_{iv}^*(x)$  and thus it becomes trivial to distinguish between the function families  $\mathcal{H}_f$  and  $\mathcal{H}_{all}$ .<sup>2</sup> Hence, the inequality (6.9) fails for a general class of adversaries that access functions  $f_{iv}^*$  and  $f$  in a black-box manner. In more explicit terms, there might exist a universal black-box algorithm  $\mathcal{A}$  that breaks the construction for all possible iterative hash functions. Hence, one should use a one-way function  $g$  to get the final hash value  $h(x) = g(f_{iv}^*(x_1, \dots, x_n))$  in order to reduce the amount of subjective trust.

## 6.4 SETUP ASSUMPTIONS AND SUBJECTIVE SECURITY

In a certain sense, it is impossible to avoid subjective security. Even if we try to prove the security in the objective non-uniform computational model, we soon reach fundamental limitations of hypothesis testing. Many cryptographic protocols are specified as a part of a communication standard, such as the DSS standard [Nat00] for digital signatures, or the TLS v1.1 standard [Net06] for secure web browsing. Such standards almost always explicitly fix some cryptographic primitives, for example, state that messages must be hashed with SHA-1. Hence, contrary to the belief of most theoreticians, almost all cryptographic applications are implemented in the *common reference string model*, where all protocol participants can access authentically distributed setup parameters.

Such a setting is formally modelled by a *trusted setup procedure*, where a non-corruptible party  $\mathcal{P}_{ts}$  sends some initial parameters to the participants of the protocol. Usually, it is sufficient if the communication between the dealer  $\mathcal{P}_{ts}$  and other participants is authentic but some settings also require confidentiality.

<sup>2</sup>The argumentation can be generalised for the actual hash functions with a padding.

For example, the Digital Signature Standard fixes some recommended parameters including SHA-1 as a preferred hash function. Another class of similar examples consists of various theoretical models for public-key infrastructure.

From a technical viewpoint, analysis of a protocol with a trusted setup procedure is not different from ordinary protocol, except that the real world model has an extra phase that is missing in the ideal world, see Fig. 6.4. A bigger, not to say an essential, problem lies in the fact that the corresponding objective security model does not faithfully reflect the reality:

- For many standards, the setup procedure is run only once. Consequently, one really cares how successful a potential adversary is for this *particular* outcome of the setup procedure, whereas the objective security finds only an *average-case* success bound over all possible runs of the setup.
- Public setup parameters  $\text{psp}$  are often fixed for years. Thus, an adversary can and should *in principle* devise a specific attack that utilises the properties of  $\text{psp}$ . No objective security model can capture such a dependence, as the algorithms are always chosen *before* the parameters are generated.

We emphasise that the problem lies in the fundamental mismatch between individual and collective properties of the setup procedure and thus cannot be resolved by clever modifications of security games.

It is instructive to consider a couple of concrete examples before giving more abstract reasoning. Recall that a function family  $\mathcal{H}$  is  $(t, \varepsilon)$ -collision resistant if

$$\text{Adv}_{\mathcal{H}}^{\text{cr}}(\mathcal{A}) = \Pr [h \leftarrow \mathcal{H}, (x_0, x_1) \leftarrow \mathcal{A}(h) : h(x_0) = h(x_1) \wedge x_0 \neq x_1] \leq \varepsilon$$

for any  $t$ -time adversary  $\mathcal{A}$ . Observe that collision resistance is a collective property of the function family  $\mathcal{H}$ . For a fixed non-injective function  $h$ , the notion of collision resistance does not make sense, since there exists a trivial adversary  $\mathcal{A}_{x_0, x_1}$  that outputs a fixed collision  $h(x_0) = h(x_1)$ . Such discrepancy between individual and collective properties leads to a hashing paradox:

- It is impossible to *objectively* use collision resistant hash functions, even if one has an explicit description of a collision resistant function family  $\mathcal{H}$ .
- Any protocol that is compromised by a hash collision  $h(x_0) = h(x_1)$  becomes insecure in the *objective* sense as soon as  $h$  is chosen from  $\mathcal{H}$ , as we can then use a fixed algorithm  $\mathcal{A}_{x_0, x_1}$  to break the protocol.

The problem is not specific to any practical hash function, such as SHA-1 or WHIRLPOOL, nor to the way they have been designed. The hashing paradox holds for any imaginable compressing function. In particular, it is impossible to talk about the *objective security* of a digital signature scheme that uses a fixed hash function to compress messages before signing them.

As a second example, we present a similar paradox for asymmetric encryption. Observe that IND-CPA security is again a collective property of a cryptosystem. For a fixed public key  $\text{pk}$  there exists a trivial adversary  $\mathcal{A}_{\text{sk}}$  that uses a hard-wired secret key  $\text{sk}$  to successfully decrypt all messages encrypted by  $\text{Enc}_{\text{pk}}(\cdot)$ . Consequently, we can state an encryption paradox:

- It is impossible to use an *objectively* confidential asymmetric enciphering method even if one has explicit description of an IND-CPA cryptosystem.

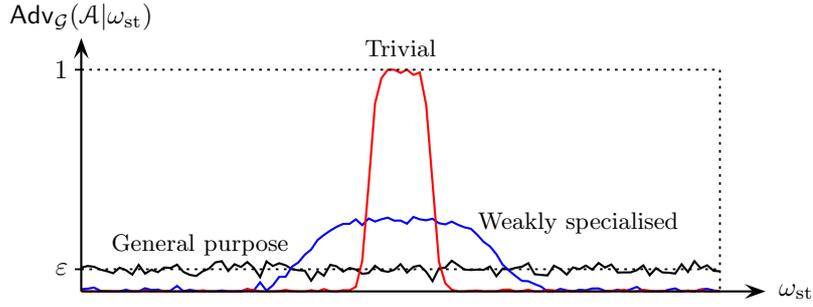


Figure 6.5: The advantage  $\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}})$  as a function of randomness  $\omega_{\text{ts}}$  used by the dealer  $\mathcal{P}_{\text{ts}}$  for various different types of attacking algorithms.

- Any protocol that uses asymmetric encryption becomes insecure in the *objective sense* as soon as the public key  $\text{pk}$  is officially published.

A similar paradox emerges for digital signatures. Moreover, this problem cannot be ignored as a weird theoretical observation that has no practical consequences, since the public-key infrastructures are common in everyday life. We have enjoyed secure web-browsing from 1994, when Netscape developed the first version of SSL protocol, and many countries are deploying country-wide official public-key infrastructures. All such infrastructures collapse if one obtains a master secret key that is used to guarantee the authenticity of all public keys. Hence, a potential adversary needs to crack a single *fixed* master key.

For clarity, assume that the master secret key is a full factorisation of an RSA modulus  $N = p \cdot q$ . Now finding a factorisation of a specific integer  $N$  might be much easier than the factoring problem in general. For example, the state of the art factoring efforts have produced a full factorisation of a 1039-bit Mersenne number, whereas the current record of factoring RSA moduli is 640-bits [AFK<sup>+</sup>07, RSA07]. However, if an adversary can specify an algorithm for the concrete modulus  $N$ , then we cannot exclude a trivial algorithm  $\mathcal{A}_{p,q}$  that just prints out the corresponding factors  $p$  and  $q$ . Thus, no *objective* security model can capture the intractability of factoring for a particular modulus  $N$ .

The essence of the trusted setup problem is revealed, if we consider an advantage of a  $t$ -time algorithm  $\mathcal{A}$  as a function of random coins used by  $\mathcal{P}_{\text{ts}}$

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}}) = \Pr[\mathcal{G}^{\mathcal{A}} = 1 | \mathcal{P}_{\text{ts}} \text{ uses randomness } \omega_{\text{ts}}] \quad . \quad (6.13)$$

As the overall advantage can be computed as an average

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}) = \frac{1}{|\Omega_{\text{ts}}|} \cdot \sum_{\omega_{\text{ts}} \in \Omega_{\text{ts}}} \text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}}) \quad , \quad (6.14)$$

the objective security bounds limit only the area under the profile, as illustrated in Fig. 6.5. Now the advantage for a fixed setup run is just a single point in the profile of  $\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}})$ . Consequently, objective setting for a fixed setup run is meaningful only if for all  $t$ -time adversaries  $\mathcal{A}$  and randomness  $\omega_{\text{ts}} \in \Omega_{\text{ts}}$

$$\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}}) \leq \varepsilon_0 \ll 1 \quad . \quad (6.15)$$

But then the setup procedure is redundant, as we can fix all random coins  $\omega_{\text{ts}} = 0 \dots 0$  and let all parties compute the setup messages by themselves. Analogously, we can remove the setup procedure even if the inequality (6.15) holds only for a single choice of random coins  $\omega_{\text{ts}}^* \in \Omega_{\text{ts}}$ .

Hence, for any *meaningful* setup procedure, there must exist a trivial attack for any protocol run. We can eliminate them, but the corresponding choice between weakly specialised and trivial attacks is inherently *subjective*. Fortunately, there is a *heuristic* bridge between objective and subjective security.

**Fairness Postulate.** *If the setup procedure has been properly carried out, it is rational to assume that the value  $\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}})$  exceeds an objective bound  $\text{Adv}_{\mathcal{G}}(\mathcal{A}) \leq \varepsilon$  only by several magnitudes for all relevant algorithms  $\mathcal{A} \in \mathfrak{A}$ .*

Of course, it is impossible to prove the postulate, we can only justify it. Since the area under the profile is bounded by  $\varepsilon$ , the advantage  $\text{Adv}_{\mathcal{G}}(\mathcal{A}|\omega_{\text{ts}})$  can exceed  $c \cdot \varepsilon$  only for a fraction  $\frac{1}{c}$  of all  $\omega_{\text{ts}} \in \Omega_{\text{ts}}$ . Hence, algorithms with ultra-high advantages must be extremely specialised and should be left out. Also, if an attacker manages to find an ultra-specialised algorithm  $\mathcal{A}$  for a significant fraction of  $\omega_{\text{ts}} \in \Omega_{\text{ts}}$ , then the attacker itself as a self-tuning attacking strategy exceeds the objective security bound. Evidently, we cannot exclude such super-intelligent adversaries but in that case there is nothing we could do anyway.

Finally, we emphasise that objective and subjective security are not conflicting concepts but rather complementary formalisations. Objective security provides a rigorous way to design protocols that are resistant to general purpose attacks, whereas subjective security is unavoidable if we want to analyse security guarantees after some parameters are fixed. A similar duality between the design and the actual usage also appears in the statistics. It is possible to design statistical estimation algorithms that behave well on average, but a meaningful interpretation of the resulting outputs must use subjective probabilities.

## 6.5 RIGOROUS FORMALISATION OF SUBJECTIVE SECURITY

The fairness postulate has the same drawbacks as the subjective equivalence assumption used in the random oracle model. Both of them rely on inherently subjective decisions, which might turn out to be inappropriate in retrospection. Therefore, one should apply them only in very simplistic settings and use the resulting subjective security premises as stepping stones in the analysis of more complex constructions. Such an approach reduces the amount of subjectivity by locating the essential subjective security premises one must believe.

A proper formalisation of subjective security premises and proofs is needed for other more technical reasons as well. First, it is not evident at all that excessive application of the fairness postulate or its analogues does not lead to inconsistencies. Moreover, there are no objectively secure cryptographic primitives known so far and thus the fairness postulate is *de facto* inapplicable. Additionally, it is not apparent at all that compact direct proofs do exist in the computational setting, even if primitives like one-way functions exist. In other words, we do not know whether one can ever provide a complete security proof for a specific cryptographic primitive that is verifiable in reasonable time. As a result, the subjective security might be truly unavoidable. Finally, a proper formalisation of

subjective security is interesting in its own right, as it clarifies what we actually mean by subjective security. In particular, what is the main difference between security proofs in objective and subjective settings.

It turns out that most proofs can be freely translated from the objective setting to the subjective setting and vice versa, and that the main difference is only in the exact bounds on the running times. The latter is somewhat expected, as the uniform polynomial security model is also subjective and so far most of the published security proofs hold for both polynomial security models.

**Formalisation of subjective security.** At first glance, it seems impossible to formalise the intuition that some algorithms are not accessible to adversary, since we do not know how the algorithms are generated. Indeed, we showed above that such a formalisation is impossible in the classical frequentistic setting. However, if we accept the subjectivity of probabilities, the solution becomes evident. Namely, we must assign *subjective occurrence properties* to all possible adversarial algorithms to exclude ultra-specialised attacks.

More formally, one must specify a distribution of algorithms  $\mathfrak{A}$  for a particular problem. A choice of the corresponding distribution is entirely subjective, as it reflects personal beliefs on which algorithms are more likely to be used in potential attacks. In practice, one does not have to specify the entire distribution, it is sufficient to fix several *unquestionable beliefs* about adversarial distribution and then use principles of coherent reasoning. A fully determined distribution  $\mathfrak{A}$  is needed to formally define a *relative advantage against a game  $\mathcal{G}$*

$$\text{Adv}_{\mathcal{G}}(\mathfrak{A}) = \Pr[\mathcal{A} \leftarrow \mathfrak{A} : \mathcal{G}^{\mathcal{A}} = 1] \quad (6.16)$$

and a *relative distinguishing advantage for a game pair  $\mathcal{G}_0$  and  $\mathcal{G}_1$*

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathfrak{A}) = |\Pr[\mathcal{A} \leftarrow \mathfrak{A} : \mathcal{G}_0^{\mathcal{A}} = 1] - \Pr[\mathcal{A} \leftarrow \mathfrak{A} : \mathcal{G}_1^{\mathcal{A}} = 1]| \quad (6.17)$$

We emphasise that the distribution  $\mathfrak{A}$  does not have to be not uniform. Hence, the success probabilities of more probable algorithms  $\mathcal{A}$  have a much bigger impact to  $\text{Adv}_{\mathcal{G}}(\mathfrak{A})$  and  $\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathfrak{A})$  than other less probable algorithms.<sup>3</sup>

Observe that it is reasonable to specify different distributions for different time bounds  $t$ , as some attacks might require a certain amount of time to be useful at all. Hence, we use a shorthand  $\mathfrak{A}(\cdot)$  to denote a family of distributions, where  $\mathfrak{A}(t)$  denotes a distribution of  $t$ -time adversaries. Now a *relative computational distance with respect to a time bound  $t$*  is defined as

$$\text{cd}_{\star}^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_1) = \text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathfrak{A}(t)) \quad (6.18)$$

As an example, recall that in the uniform polynomial model, a distribution family  $\mathfrak{A}(\cdot)$  consists of a single adversary  $\mathcal{A}$  that gets a security parameter  $k$  as an extra argument. Also, note that even a non-uniform adversary can be modelled by a distribution family, where  $\mathfrak{A}(t)$  consists of a single adversary  $\mathcal{A}_t$  that is optimal over the set of all  $t$ -time adversaries.

Now, as the distribution family  $\mathfrak{A}(\cdot)$  puts restrictions on the set of plausible adversaries, the security with respect to fixed parameters becomes meaningful.

<sup>3</sup>Of course, there are no objective measures for determining the right occurrence probabilities for all algorithms, since we cannot objectively predict what “useful” algorithms will be discovered and used in the future. Hence, these notions are inherently subjective.

In particular, the following instantiations of IND-CPA games

$$\begin{array}{l} \mathcal{Q}_0^A \\ \left[ \begin{array}{l} (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_0) \\ \text{return } \mathcal{A}(c) \end{array} \right. \end{array} \quad \begin{array}{l} \mathcal{Q}_1^A \\ \left[ \begin{array}{l} (m_0, m_1) \leftarrow \mathcal{A}(\text{pk}) \\ c \leftarrow \text{Enc}_{\text{pk}}(m_1) \\ \text{return } \mathcal{A}(c) \end{array} \right. \end{array}$$

where  $\text{pk}$  is a fixed valid public key, might lead to nontrivial bounds on the relative computational distance  $\text{cd}_*^{\mathcal{Q}(t)}(\mathcal{Q}_0, \mathcal{Q}_1)$ . In other words, the subjective notion of IND-CPA security is meaningful also for a fixed public key  $\text{pk}$ .

**Principles of coherent reasoning.** Since a rational entity is free to choose subjective probabilities, it is impossible to formally prove the (in)validity of security assumptions. We can only try to eliminate inconsistencies between various beliefs in order to achieve coherence. Therefore, we are going to mimic the classical approach that is used to formalise subjective probabilities, see handbooks [Jay03, Jef04]. In a nutshell, we assume that all subjective probability distributions are assigned by a *rational entity* who is willing to correct his or her assignments if he or she finds out that they contradict common sense. We specify this vague definition by fixing a set of axioms (R1)–(R3) that directly follow from the common-sense understanding of attack strategies.

For brevity, we only consider probability assignments for game pairs, since a relative advantage  $\text{Adv}_{\mathcal{G}}(\mathfrak{A})$  can be restated as a relative computational distance  $\text{Adv}_{\mathcal{G}, \perp}^{\text{ind}}(\mathfrak{A})$ , where  $\perp$  denotes a game that always ends with  $\perp$ . Assume that a rational entity has specified different adversarial distributions  $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \dots$  for different problems. In other words, the rational entity acknowledges that an attacker can sample an algorithm from these *candidate distributions*  $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}, \dots$ . Then it is irrational to assume that a potential attacker cannot use the candidate distribution  $\mathfrak{A}$  for other problems. Similarly, it is irrational to assume that an attacker cannot use a well-known code transformation  $\mathcal{T}$  to convert a sampled adversarial algorithm  $\mathcal{A}$  to a different algorithm  $\mathfrak{B} \leftarrow \mathcal{T}(\mathcal{A})$ . For example, there is no reason to refute the possibility that an attacker may encode a factoring task as a Boolean formula if he or she has access to a very efficient solver for Boolean formulae. In other terms, well-known transformations introduce new candidate distributions  $\mathcal{T}(\mathfrak{A}), \mathcal{T}(\mathfrak{B}), \mathcal{T}(\mathfrak{C}), \dots$ . However, such transformations are not for free, since the application of the transformation itself takes time. Shortly, if  $\mathfrak{B} \leftarrow \mathcal{T}(\mathcal{A})$ , then the time needed to compute  $\mathcal{T}(\mathcal{A})$  is added to the running time of  $\mathfrak{B}$ . As a result, we have discovered three basic coherence axioms that reveal inconsistencies between subjective probability assignments:

- (R1) Let  $\mathcal{T}$  be a code transformation *known* to a rational entity and  $\mathfrak{A}$  be a candidate distribution for distinguishing  $\mathcal{G}_0$  and  $\mathcal{G}_1$ . Then a distribution  $\mathfrak{B} = \mathcal{T}(\mathfrak{A})$  induced by  $\mathcal{T}$  and  $\mathfrak{A}$  must also be a candidate distribution.
- (R2) Let  $\mathfrak{B} = \mathcal{T}(\mathfrak{A})$  as specified above. Then the time needed to compute  $\mathcal{T}(\mathcal{A})$  from a sample  $\mathcal{A} \leftarrow \mathfrak{A}$  must be added to the running time of  $\mathcal{T}(\mathcal{A})$ .
- (R3) Let  $\mathfrak{A}(t)$  and  $\mathfrak{B}(t)$  be two candidate distributions for distinguishing  $\mathcal{G}_0$  and  $\mathcal{G}_1$ . Now if the entity *knows* that  $\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_1) < \text{cd}_*^{\mathfrak{B}(t)}(\mathcal{G}_0, \mathcal{G}_1)$  then the entity must prefer the distribution  $\mathfrak{B}(t)$  to the distribution  $\mathfrak{A}(t)$ .

The verb ‘know’ and the term ‘knowledge’ are used here to denote explicit awareness, i.e., an entity who reasons about security must literally know the

description of a transformation  $\mathcal{T}$  or possess a valid proof that the inequality  $\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_1) < \text{cd}_*^{\mathfrak{B}(t)}(\mathcal{G}_0, \mathcal{G}_1)$  holds. In particular, a person who knows nothing about cryptography can arbitrarily assign adversarial distributions to various problems, since any assignment is coherent with his or her knowledge about these problems. When the person gains more knowledge, then certain assignments become irrational and thus he or she must sometimes accept that certain constructions are secure. Nevertheless, note that a set of facts rarely determines unique candidate distributions for all problems and thus entities can reach different conclusions even if they share the same set of facts. In other words, a rational entity can and should discover inconsistencies according to his or her knowledge, but a certain amount of subjectivity is unavoidable in his or her assignments.

Note that all security proofs must be strongly constructive in the subjective setting, since a rational entity must explicitly know a distribution  $\mathfrak{B}$  that is a better alternative than a distribution  $\mathfrak{A}$ . A mere existence of a better alternative does not cause inconsistencies as long as the rational entity is not informed. Such a position is unavoidable, or otherwise we cannot escape the objective setting. If we require consistency for all possible transformations, then we also allow trivial transformations that ignore the input code and output an optimal adversary for a particular problem and we back in the objective setting.

Finally, observe that the axiomatisation cannot lead to contradictory restrictions for adversarial distributions as long as the objective security model itself is non-contradictory. Namely, a rational entity can always assume that a distribution  $\mathfrak{A}(t)$  for a game pair  $\mathcal{G}_0$  and  $\mathcal{G}_1$  consist of a single  $t$ -time adversary  $\mathcal{A}$  that maximises the advantage  $\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A})$ . Consequently, there can be no better alternatives for the distribution  $\mathfrak{A}(t)$  and none of the axioms (R1)–(R3) is violated. Note that this conservative assignment of adversarial distributions corresponds to the objective setting and thus must be well defined.

## 6.6 REDUCTIONS AND SUBJECTIVE SECURITY PREMISES

The axioms of coherent reasoning (R1)–(R3) alone are not sufficient to prove the subjective security of a protocol or a construction, as they just reveal the inconsistencies. A decision how such inconsistent assignments should be changed is also a subjective decision and can depend on personal preferences of rational entities. To solve the ambiguity, one must fix a small set of *fundamental beliefs* that is never changed if it is possible to achieve coherence otherwise. These beliefs will play the role of subjective security assumptions. Formally, a *basic security premise*  $\llbracket \mathcal{Q}_0 \cong \mathcal{Q}_1 | (t, \varepsilon) \rrbracket$  is an unquestionable belief that the relative computational distance satisfies  $\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{Q}_0, \mathcal{Q}_1) \leq \varepsilon$  for any achievable distribution family  $\mathfrak{A}(\cdot)$ . As a result, we can formalise the subjective  $(t, \varepsilon)$ -collision resistance for a function  $h$  and  $(t, \varepsilon)$ -IND-CPA security for a public key  $\text{pk}$  by adding the corresponding security premises into the set of fundamental beliefs.

We emphasise here that fundamental beliefs are empirically falsifiable. For example, the MD5 hash function was believed to be collision resistant in 1992 but the publication of successful attacks has forced us to change our beliefs.

Technically, subjective security proofs are very similar to classical proofs. Assume that a security premise  $\llbracket \mathcal{Q}_0 \cong \mathcal{Q}_1 | (t_0, \varepsilon_0) \rrbracket$  is fundamental. Then an effi-

cient constructive reduction  $\mathcal{T}$  that maps an algorithm  $\mathcal{A}$  to  $\mathcal{B}$  so that

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A}) > \varepsilon \quad \Longrightarrow \quad \text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathcal{B}) > \rho(\varepsilon) \quad (6.19)$$

for a known function  $\rho : [0, 1] \rightarrow [0, 1]$  introduces an upper bound on the computational distance  $\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_1)$ . Namely, if the security guarantee  $\rho$  is convex-cup, the corresponding distribution  $\mathfrak{B} = \mathcal{T}(\mathfrak{A})$  satisfies

$$\text{cd}_*^{\mathfrak{B}}(\mathcal{Q}_0, \mathcal{Q}_1) = \sum_{\mathcal{A} \in \mathfrak{A}} \Pr[\mathcal{A}] \cdot \text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathcal{T}(\mathcal{A})) \geq \rho(\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_1)) \quad , \quad (6.20)$$

as the definition of  $\rho$  and Jensen's inequality assure

$$\sum_{\mathcal{A} \in \mathfrak{A}} \Pr[\mathcal{A}] \cdot \text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathcal{T}(\mathcal{A})) \geq \sum_{\mathcal{A} \in \mathfrak{A}} \Pr[\mathcal{A}] \cdot \rho(\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A})) \quad , \quad (6.21)$$

$$\sum_{\mathcal{A} \in \mathfrak{A}} \Pr[\mathcal{A}] \cdot \rho(\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A})) \geq \rho\left(\sum_{\mathcal{A} \in \mathfrak{A}} \Pr[\mathcal{A}] \cdot \text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathcal{A})\right) \quad . \quad (6.22)$$

Therefore, we obtain an average-case security guarantee

$$\text{Adv}_{\mathcal{G}_0, \mathcal{G}_1}^{\text{ind}}(\mathfrak{A}) > \varepsilon \quad \Longrightarrow \quad \text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathfrak{B}) > \rho(\varepsilon) \quad . \quad (6.23)$$

If the security guarantee  $\rho$  is not convex-cup, the pointwise guarantee (6.19) does not automatically imply the average-case guarantee (6.23) and we have to bound  $\text{Adv}_{\mathcal{Q}_0, \mathcal{Q}_1}^{\text{ind}}(\mathfrak{B})$  by using other methods. Given a fundamental security premise  $\llbracket \mathcal{Q}_0 \cong \mathcal{Q}_1 \rrbracket(t_0, \varepsilon_0)$  and an average case security guarantee (6.23) that respects the time bound  $t_0$ , we can conclude

$$\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_1) \leq \rho^{-1}(\varepsilon_0) \quad , \quad (6.24)$$

otherwise we violate either the axiom (R3) or the security premise. Hence, subjective security proofs are technically identical to traditional objective security proofs, provided that the code transformation  $\mathcal{T}$  has an explicit description. Moreover, as all common pointwise reductions have convex-cup security guarantees, and even the quantitative success bounds of traditional and subjective security estimates coincide. The difference appears only in running times, since we must also consider the complexity of the code transformation  $\mathcal{T}$ .

We remark here that the same issues have also been addressed by Rogaway who tried to solve the hashing paradox described in Section 6.4. However, the corresponding article [Rog06] does not provide a proper mathematical foundation of subjective security (human ignorance). In fact, Rogaway believes that rigorous formalisation of subjective security is impossible [Rog06]:

*What is meant is that there is no efficient algorithm known to man that outputs a collision in  $H$ . But such a statement would seem to be unformalizable—outside the realm of mathematics.*

We have clearly shown that such a formalisation is possible as soon as one is willing to accept subjective interpretation of probabilities. Moreover, our treatment of subjective security also provides a formal justification to the methodology used in [Rog06]. More precisely, Rogaway required that all reductions

should be strictly constructive and accompanied with appropriate pointwise security guarantees (6.19). As a result, both formalisations give qualitatively equivalent security guarantees for most constructions and protocols.

However, there is still a major conceptual difference that makes our formalisation different from the traditional objective setting and the formalisation given by Rogaway. Namely, the axioms (R1)–(R3) are applicable only if a *concrete* security premise is *known* to be violated. Therefore, we cannot prove

$$\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_2) \leq \varepsilon_1 + \varepsilon_2 \quad (6.25)$$

from premises  $\llbracket \mathcal{G}_0 \cong \mathcal{G}_1 | (t, \varepsilon_1) \rrbracket$  and  $\llbracket \mathcal{G}_1 \cong \mathcal{G}_2 | (t, \varepsilon_2) \rrbracket$ , although the violation of the inequality (6.25) implies that one of the inequalities

$$\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_1) > \varepsilon_1 \quad \text{or} \quad \text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_1, \mathcal{G}_2) > \varepsilon_2 \quad (6.26)$$

holds. The problem lies in the fact that we *do not know* which disjunct is true and thus the distribution  $\mathfrak{A}(t)$  that violates the inequality (6.25) does not contradict the axioms (R1)–(R3) nor security premises. Hence, a non-constructive use of triangle inequality is not allowed. One might think that such a strong constructivity requirement is artificial and redundant. However, a consistent formalisation of subjective security without this assumption is impossible, otherwise we end up in the objective setting. For example, we can always consider singleton distributions  $\mathfrak{A}_i(t)$  that consist of a single  $t$ -time algorithm  $\mathcal{A}_i$ . Now if  $\mathfrak{A}_1(t), \dots, \mathfrak{A}_s(t)$  cover all  $t$ -time algorithms, the formula

$$\text{cd}_*^{\mathfrak{A}_1(t)}(\mathcal{G}_0, \mathcal{G}_1) > \varepsilon \vee \text{cd}_*^{\mathfrak{A}_2(t)}(\mathcal{G}_0, \mathcal{G}_1) > \varepsilon \vee \dots \vee \text{cd}_*^{\mathfrak{A}_s(t)}(\mathcal{G}_0, \mathcal{G}_1) > \varepsilon \quad (6.27)$$

holds whenever  $\varepsilon < \text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1)$ . Hence, we cannot conclude the contradiction from the formula (6.27), or otherwise the non-contradictory assignment must satisfy

$$\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_1) = \text{cd}_*^t(\mathcal{G}_0, \mathcal{G}_1) . \quad (6.28)$$

Since formulae (6.26) and (6.27) are analogous, we must treat them similarly.

In many occasions, we can circumvent the problem by finding more explicit reductions, but when a construction is based on different basic primitives, it is often impossible to eliminate non-constructivity. Therefore, we allow explicit strengthenings of fundamental beliefs. More formally, let

$$\llbracket \mathcal{Q}_0 \cong \mathcal{Q}_1 | (t_1, \varepsilon_1) \rrbracket \wedge \llbracket \mathcal{Q}_2 \cong \mathcal{Q}_3 | (t_2, \varepsilon_2) \rrbracket \quad (6.29)$$

denote a *strengthened premise* that distributions  $\mathfrak{A}$  and  $\mathfrak{B}$  are inconsistent if

$$\text{cd}_*^{\mathfrak{A}(t_1)}(\mathcal{Q}_0, \mathcal{Q}_1) > \varepsilon_1 \quad \text{or} \quad \text{cd}_*^{\mathfrak{B}(t_2)}(\mathcal{Q}_2, \mathcal{Q}_3) > \varepsilon_2 \quad (6.30)$$

even if it is not known which disjunct of the statement holds. Such explicit way of strengthening keeps the amount of non-constructiveness under tight control and we do not lose the connection with theoretical foundations.

Moreover, it is possible to express strengthened beliefs as a belief about a special game if we use a signed advantage

$$\overline{\text{cd}}_*^{\mathfrak{A}(t_1)}(\mathcal{G}_0, \mathcal{G}_1) = \Pr[A \leftarrow \mathfrak{A} : \mathcal{G}_0^A = 1] - \Pr[A \leftarrow \mathfrak{A} : \mathcal{G}_1^A = 1] \quad (6.31)$$

instead of the computational distance  $\text{cd}_*^{\mathfrak{A}(t_1)}(\mathcal{G}_0, \mathcal{G}_1)$ . We omit the corresponding technical details, since it is just a theoretically pleasing reassurance that the axioms (R1)–(R3) are expressive enough and does not give additional insight.

**Subjective security for protocols.** Recall that the classical ideal real world comparison was just a convenient way to define security with respect to all relevant security goals  $\mathfrak{B}$ . More precisely, the classical definition allowed to replace the real world adversary  $\mathcal{A}$  with an ideal world adversary  $\mathcal{A}^\circ$  in order to get a contradiction. The subjective security setting also adds an efficiency restriction. The real and ideal world are *constructively*  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t. a *distribution of adversaries*  $\mathfrak{A}$  and a *distribution of security goals*  $\mathfrak{B}$  if there exists a code transformation  $\mathcal{T}$  that satisfies the following constraints. First,  $\mathcal{T}$  transforms a distribution of real world adversaries  $\mathfrak{A}$  into a distribution of ideal world adversaries  $\mathfrak{A}_\circ$ . Secondly, the time needed to compute  $\mathcal{A}^\circ = \mathcal{T}(\mathcal{A})$  is included in the running time of  $\mathcal{A}^\circ$ . Finally, for the time bounds  $t_{\text{re}}$  and  $t_{\text{id}}$  on the running times of  $\mathcal{A}$  and  $\mathcal{A}^\circ$  and for any feasible input distribution  $\mathfrak{D}$ :

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon, \quad (6.32)$$

where the probability is also taken over the distributions  $\mathfrak{A}$ ,  $\mathfrak{A}_\circ$  and  $\mathfrak{B}$ .

Observe that we must limit the set of security goals  $\mathfrak{B}$  that are used to compute the outputs of  $\mathcal{G}_{\text{real}}$  and  $\mathcal{G}_{\text{ideal}}$  or otherwise the requirements may be too strict. For example, consider a protocol where an honest party  $\mathcal{P}_1$  sends an encryption  $\text{Enc}_{\text{pk}}(x_1)$  to corrupted  $\mathcal{P}_2$ . If  $\text{pk}$  is a fixed public parameter, then there exists an efficient predicate  $\mathfrak{B}$  that uses hardwired  $\text{sk}$  to decrypt the output of  $\mathcal{A}$ . Hence a trivial adversary  $\mathcal{A}$  that outputs  $\text{Enc}_{\text{pk}}(x_1)$  clearly violates input-privacy. To put it in another way, the distribution  $\mathfrak{B}$  must be restricted as it captures also the further offline behaviour of an adversary.

Now assume that one has provided a universal simulator construction  $\text{Sim}$  and shown that a  $t_{\text{re}}$ -time distribution  $\mathfrak{A}$  and  $t_{\text{pr}}$ -time distribution  $\mathfrak{B}$  that violate inequality (6.32) for some input distribution  $\mathfrak{D}$  can be used to construct an adversary distribution that violates a subjective security premise. Then for all coherent distributions  $\mathfrak{A}$ ,  $\mathfrak{B}$  and  $\mathfrak{D}$  the reduction schema

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon]{\text{MPC-SEC}} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \quad (6.33)$$

still holds and the real and the ideal worlds are *subjectively*  $(t_{\text{re}}, t_{\text{id}}, t_{\text{pr}}, \varepsilon)$ -close w.r.t. *the security premises*. To summarise, the proximity is still defined by the reduction schema (6.33) but the latter may fail for incoherent distributions.

Consider the uniform polynomial security model as an illustrative example. Evidently, not all input distributions are allowed or otherwise an adversary can use the inputs of corrupted parties as an external advice and we are back in the non-uniform model. Hence, all inputs should be generated by uniform polynomial algorithm. Also, note that the security goal  $\mathfrak{B}(\cdot)$  must be a uniform algorithm or otherwise reductions may fail, too.

## 6.7 STRICTLY CONSTRUCTIVE PROOF TECHNIQUES

Classical security proofs can be invalid in the context of subjective security, since they may contain non-constructive steps. In particular, one cannot use game

trees in the classical manner, as an application of a triangle inequality or a horizon splitting is a non-constructive proof step. Fortunately, most non-constructive proof steps can be substituted by constructive counterparts.

Let us start with the constructive hybrid argument. The technique itself is quite old and can be traced back to the early works of Yao and Goldreich [Yao82, Gol98]. Namely, consider a classical non-constructive game chain

$$\begin{array}{ccccccc}
\mathcal{G}_0 \Longrightarrow \mathcal{G}_1 & & \mathcal{G}_1 \Longrightarrow \mathcal{G}_2 & & \cdots & & \mathcal{G}_{n-1} \Longrightarrow \mathcal{G}_n \\
\downarrow \mathcal{T}_1 & & \downarrow \mathcal{T}_2 & & & & \downarrow \mathcal{T}_n \\
\mathcal{Q}_{00}, \mathcal{Q}_{01} & & \mathcal{Q}_{10}, \mathcal{Q}_{11} & & \cdots & & \mathcal{Q}_{n-1,0}, \mathcal{Q}_{n-1,1}
\end{array}$$

where each game has an efficient reduction  $\mathcal{T}_i$  to another more elementary game pair  $\mathcal{Q}_{i,0}$  and  $\mathcal{Q}_{i,1}$  and the reduction provides perfect simulation

$$\mathcal{Q}_{i,0}^{\mathcal{T}_i(\mathcal{A})} \equiv \mathcal{G}_{i-1}^{\mathcal{A}} \quad \text{and} \quad \mathcal{Q}_{i,1}^{\mathcal{T}_i(\mathcal{A})} \equiv \mathcal{G}_i^{\mathcal{A}} . \quad (6.34)$$

Then we can express  $\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_n)$  as a telescopic sum

$$\left| \sum_{\mathcal{A} \in \mathfrak{A}} \sum_{i=1}^n \Pr[\mathcal{A}] \cdot \Pr[\mathcal{Q}_{i,0}^{\mathcal{T}_i(\mathcal{A})} = 1] - \sum_{\mathcal{A} \in \mathfrak{A}} \sum_{i=1}^n \Pr[\mathcal{A}] \cdot \Pr[\mathcal{Q}_{i,1}^{\mathcal{T}_i(\mathcal{A})} = 1] \right| . \quad (6.35)$$

In the simplest case, all elementary game pairs coincide  $(\mathcal{Q}_{i,0}, \mathcal{Q}_{i,1}) \equiv (\mathcal{Q}_0, \mathcal{Q}_1)$ . A randomised transformation that given a code sample  $\mathcal{A}$  from  $\mathfrak{A}$  applies the transformation  $\mathcal{T}_i$  with probability  $\frac{1}{n}$  produces a distribution  $\mathfrak{B}$  such that

$$\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_n) \leq n \cdot \text{cd}_*^{\mathfrak{B}}(\mathcal{Q}_0, \mathcal{Q}_1) , \quad (6.36)$$

since the term  $n \cdot \text{cd}_*^{\mathfrak{B}}(\mathcal{Q}_0, \mathcal{Q}_1)$  can be also expressed as the sum (6.35). Generally, we have  $n_j$  different transformations  $\mathcal{T}_i$  that lead to the same game pairs  $(\mathcal{Q}_{2j-2}, \mathcal{Q}_{2j-1})$ . Now for each  $j \in \{1, \dots, k\}$ , we can generate an adversarial distribution  $\mathfrak{B}_j$  by choosing uniformly at random a transformation  $\mathcal{T}_i$  that leads to the game pair  $(\mathcal{Q}_{2j-2}, \mathcal{Q}_{2j-1})$ . The resulting adversarial distributions  $\mathfrak{B}_1, \dots, \mathfrak{B}_k$  satisfy the inequality

$$\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_n) \leq n_1 \cdot \text{cd}_*^{\mathfrak{B}_1}(\mathcal{Q}_0, \mathcal{Q}_1) + \cdots + n_k \cdot \text{cd}_*^{\mathfrak{B}_k}(\mathcal{Q}_{2k-2}, \mathcal{Q}_{2k-1}) , \quad (6.37)$$

since the right hand side can again be lower bounded by the sum (6.35).

To complete the picture, we also consider a non-constructive horizon splitting step for  $\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_1)$  under the analogous simulatability assumption

$$\mathcal{Q}_{i,0}^{\mathcal{T}_i(\mathcal{A})} \equiv (\mathcal{G}_0 | \mathcal{H}_i)^{\mathcal{A}} \quad \text{and} \quad \mathcal{Q}_{i,1}^{\mathcal{T}_i(\mathcal{A})} \equiv (\mathcal{G}_1 | \mathcal{H}_i)^{\mathcal{A}} . \quad (6.38)$$

Again, we can express  $\text{cd}_*^{\mathfrak{A}(t)}(\mathcal{G}_0, \mathcal{G}_n)$  as a telescopic sum (6.35) and use similar regrouping techniques to obtain

$$\text{cd}_*^{\mathfrak{A}}(\mathcal{G}_0, \mathcal{G}_n) \leq n_1 \cdot \text{cd}_*^{\mathfrak{B}_1}(\mathcal{Q}_0, \mathcal{Q}_1) + \cdots + n_k \cdot \text{cd}_*^{\mathfrak{B}_k}(\mathcal{Q}_{2k-2}, \mathcal{Q}_{2k-1}) , \quad (6.39)$$

where the distributions  $\mathfrak{B}_1, \dots, \mathfrak{B}_k$  are generated by analogous uniform choice over the transformations corresponding to the same game pair.

To summarise, under the perfect simulation assumptions (6.34) and (6.38), we can use both game chains and horizon splitting to bound the computational distance  $\text{cd}_*^{\mathfrak{Q}(t)}(\mathcal{G}_0, \mathcal{G}_n)$  by using only a strengthened security premise

$$\llbracket \mathcal{Q}_0 \cong \mathcal{Q}_1 | (t_1, \varepsilon_1) \rrbracket \wedge \dots \wedge \llbracket \mathcal{Q}_{2k-2} \cong \mathcal{Q}_{2k-1} | (t_k, \varepsilon_k) \rrbracket .$$

Naturally, we must take into account the running times of  $\mathcal{T}_i$  and additional  $O(\log n)$  time needed for random sampling, but these are minor details.

Note that the perfect simulatability assumption is satisfied for elementary reductions that define computational indistinguishability for a primitive, for example various encryption schemes and pseudorandom generators have the corresponding reductions. Other primitives are normally defined in terms of advantage in a single game  $\mathcal{Q}_i$  like collision resistance and one-wayness. Consequently, the corresponding game hops rely on constructive reductions

$$\text{Adv}_{\mathcal{G}_{i-1}, \mathcal{G}_i}^{\text{ind}}(\mathfrak{A}) > \varepsilon_i \quad \implies \quad \text{Adv}_{\mathcal{Q}_i}(\mathcal{T}_i(\mathfrak{A})) > \rho_i(\varepsilon_i) . \quad (6.40)$$

Again, consider the simplest case where all games coincide  $\mathcal{Q}_i \equiv \mathcal{Q}$ . Then the corresponding triangle inequality

$$\text{cd}_*^{\mathfrak{Q}}(\mathcal{G}_0, \mathcal{G}_n) \leq \text{cd}_*^{\mathfrak{Q}}(\mathcal{G}_0, \mathcal{G}_1) + \dots + \text{cd}_*^{\mathfrak{Q}}(\mathcal{G}_{n-1}, \mathcal{G}_n) \quad (6.41)$$

for a game chain still holds, although we do not know the computational distances  $\varepsilon_i = \text{cd}_*^{\mathfrak{Q}}(\mathcal{G}_{i-1}, \mathcal{G}_i)$ . Nevertheless, we can get an adversarial distribution  $\mathfrak{B}$  by applying a transformation  $\mathcal{T}_i$  with uniform probability  $\frac{1}{n}$ . The corresponding advantage can be lower bounded by solving the optimisation task

$$\text{Adv}_{\mathcal{Q}}(\mathfrak{B}) \geq \min_{\varepsilon_1 + \dots + \varepsilon_n \geq \varepsilon} \frac{\rho_1(\varepsilon_1) + \dots + \rho_n(\varepsilon_n)}{n} , \quad (6.42)$$

where  $\varepsilon = \text{cd}_*^{\mathcal{A}}(\mathcal{G}_0, \mathcal{G}_n)$ . This approach can be directly generalised to handle several premises or horizon splittings. Linear bounds  $\rho_i(\cdot)$  lead us back to classical inequalities, otherwise we get slightly more loose bounds.

To summarise, most non-constructive black-box proofs can be made constructive by following the corresponding game tree. Moreover, the type of required security assumptions do not change, although the resulting time bounds and advantages may differ. The latter also applies for the white-box reductions that are efficiently constructible, i.e., the borderline between translatable and non-translatable proofs is not determined by the reduction type.

## 7 MODULAR DESIGN OF COMPLEX PROTOCOLS

Appropriate abstraction level and modular design are the key factors to success in cryptographic protocol design. In particular, it is not enough to have a definition that adequately describes the desired security goals. We also need intermediate security definitions that support modular design and analysis. More precisely, the security of a compound protocol should automatically follow from the security of its sub-protocols, or otherwise the analysis of complex protocols quickly becomes intractable. To achieve such modularity, we must consider more complex attack models than used for defining stand-alone security.

Since there are various decomposition methods, we also need protocols with different usage restrictions. In the simplest case, small sub-protocols are executed one by one to accomplish the final goal, and therefore we need a security definition that is closed under *sequential composition*. Alternatively, we can consider compound protocols that schedule sub-protocols dynamically and possibly execute them in parallel to minimise the total running time. In such cases, we need *universally composable* protocols that preserve security in such ultra-liberal settings. Of course, there are many other alternatives between these two extremes and each of them induces a new related security definition.

We emphasise that the choice of an appropriate protocol is a trade-off between various conflicting requirements and usage restrictions are an important but not the only relevant aspect. In a nutshell, various composability theorems [Ore87, Can00a, DM00, Can01] provide just a formal methodology that is needed to establish simple and natural-looking usage restrictions. Moreover, the precise knowledge of the expected usage patterns is important, as it poses structural restrictions to the corresponding protocol, see Sections 7.5–7.6.

Alternatively, we can use composability theorems for making security proofs more modular. In particular, we can use universal composability as a tool to design complex protocols in a systematic way. The corresponding methodology has three stages. First, we construct universally composable protocols for all necessary sub-tasks. More precisely, we can employ trusted setup to achieve universal composability with minimal overhead. In the second stage, we combine these sub-protocols into a round optimal solution and utilise universal composability in the corresponding security proofs. As the final design step, we replace the trusted setup phase with a corresponding sub-protocol that is secure in the stand-alone model. As a result, we obtain a round efficient protocol that is secure in the stand-alone model and has a short and modular security proof.

The formalism needed to describe and analyse various attack models is rather complicated. Hence, we start from the main concepts in Section 7.1 and only then formalise the corresponding computational model in Section 7.2. Afterwards, we gradually describe various subtle details in Sections 7.3 and 7.4 until we can establish important composability results. Last two sections are dedicated to the non-standard issues arising from our design methodology.

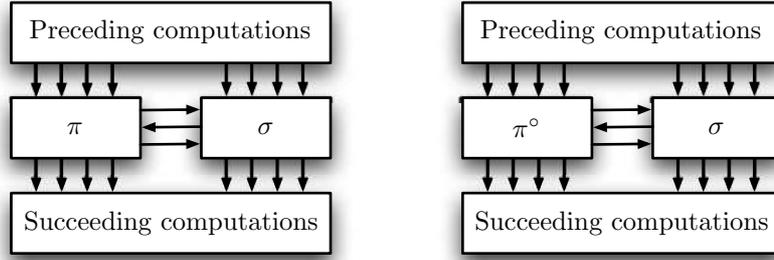


Figure 7.1: Computational context  $\varrho\langle\cdot\rangle$  is a template, which describes all computations preceding, co-occurring and following the protocols  $\pi$  and  $\pi^\circ$ .

## 7.1 DUALITY BETWEEN PROTOCOL DESIGN AND ANALYSIS

A protocol that implements a complex functionality is not only difficult to design but also difficult to analyse. To alleviate the underlying complexity, such protocols are commonly split into a collection of well-defined sub-protocols. The corresponding security proof is modular if it gradually replaces all sub-protocols with the ideal implementations until we obtain the desired ideal functionality. As a result, we have to consider the security of a protocol  $\pi$  in a *computational context*  $\varrho\langle\cdot\rangle$  that uses the protocol  $\pi$  in order to compute something else.

In the following, we always consider contexts that use protocols in a black-box manner, i.e., a context first provides inputs to the protocol and later uses the resulting outputs in subsequent computations. In fact, a context  $\varrho\langle\cdot\rangle$  is like a template that can be instantiated by specifying the missing protocol. The resulting compound protocols  $\varrho\langle\pi\rangle$  share the same general structure. Participants first precompute some values, then execute the protocol  $\pi$  together with side-computations  $\sigma$  and finally do post-processing, see Fig. 7.1 for schematic description. In particular, note that the adversarial behaviour can cause an information flow between the computational processes  $\pi$  and  $\sigma$ , although they are formally separated from each other by the specification. Such coupling effects can make the security analysis extremely difficult.

The security of a protocol  $\pi$  in a context  $\varrho\langle\cdot\rangle$  is defined by comparing the corresponding real and ideal world implementations. Let  $\varrho\langle\pi\rangle$  denote the real world implementation and  $\varrho\langle\pi^\circ\rangle$  the corresponding ideal world implementation. Then, for any input distribution  $\mathfrak{D}$  and for any security objective  $\mathcal{B}(\cdot)$ , we can define the corresponding security games

$$\begin{array}{l} \mathcal{G}_{\text{real}}^{\mathcal{A}} \\ \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathfrak{D} \\ \mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{re-atk}}^{\mathcal{A}}(\mathbf{x}) \\ \text{return } \mathcal{B}(\mathbf{z}_{\text{obs}}) \end{array} \right. \end{array} \quad \begin{array}{l} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \\ \left[ \begin{array}{l} \mathbf{x} \leftarrow \mathfrak{D} \\ \mathbf{z}_{\text{obs}} \leftarrow \mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ}(\mathbf{x}) \\ \text{return } \mathcal{B}(\mathbf{z}_{\text{obs}}) \end{array} \right. \end{array}$$

where  $\mathbf{z}_{\text{obs}}$  is the vector of observable outcomes and the sub-games  $\mathcal{G}_{\text{re-atk}}$  and  $\mathcal{G}_{\text{id-atk}}$  model online attacks against the protocols  $\varrho\langle\pi\rangle$  and  $\varrho\langle\pi^\circ\rangle$ . We omit the exact details here and discuss them separately afterwards.

More precisely, note that a context  $\varrho\langle\cdot\rangle$  defines only the usage of a protocol  $\pi$  and not the exact scheduling of protocol messages. As a result, we must formally define security in terms of game pairs, since each context might create several

game pairs  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$ . More formally, fix a set of relevant contexts  $\varrho(\cdot)$ , a set of relevant security objectives  $\mathfrak{B}$  and possible message schedulings. Let  $\mathfrak{G}$  be the set of corresponding game pairs. Then the ideal and real world implementations are  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t. all game pairs  $\mathfrak{G}$  if for any pair  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}}) \in \mathfrak{G}$  and for any  $t_{\text{re}}$ -time real world adversary  $\mathcal{A}$  there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any input distribution  $\mathfrak{D}$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (7.1)$$

Alternatively, we can view  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -closeness w.r.t.  $\mathfrak{G}$  as a reduction schema

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon]{\mathfrak{G}\text{-SEC}} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \quad (7.2)$$

that can be applied to any game pair  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}}) \in \mathfrak{G}$ . Moreover, one often defines the set of relevant contexts implicitly by specifying usage restrictions for the protocol  $\pi$ . For example, we can consider all game pairs, where the participants execute the protocol  $\pi$  sequentially or in parallel with some other protocols, i.e., we can talk about sequential and parallel composability.

In principle, we can arbitrarily choose the set of game pairs. However, only a few of these choices lead to modular security proofs. We remark that a standard design methodology creates a chain of hybrid protocols

$$\pi^\circ \equiv \varrho_0 \xrightarrow{\pi_1} \varrho_1 \xrightarrow{\pi_2} \dots \xrightarrow{\pi_{s-1}} \varrho_{s-1} \xrightarrow{\pi_s} \varrho_s \equiv \pi \quad (7.3)$$

that gradually change the ideal implementation into a practical protocol without calls to the trusted third party. More precisely, each step in the chain introduces a new sub-protocol  $\pi_i$  that implements some task needed to complete the desired functionality. Note that the sub-protocol  $\pi_i$  itself may utilise additional ideal sub-protocols that are implemented with the help of the trusted third party. These calls must be eliminated by the subsequent steps in the chain, so that the final protocol  $\pi$  does not depend on the trusted third party.

The corresponding modular security proof uses appropriate reduction schemata to traverse the chain in the opposite direction

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \equiv \mathcal{G}_s^{\mathcal{A}_s} \xrightarrow[\varepsilon_s]{\mathfrak{G}_s\text{-SEC}} \mathcal{G}_{s-1}^{\mathcal{A}_{s-1}} \xrightarrow[\varepsilon_{s-1}]{\mathfrak{G}_{s-1}\text{-SEC}} \dots \xrightarrow[\varepsilon_2]{\mathfrak{G}_2\text{-SEC}} \mathcal{G}_1^{\mathcal{A}_1} \xrightarrow[\varepsilon_1]{\mathfrak{G}_1\text{-SEC}} \mathcal{G}_0^{\mathcal{A}_0} \equiv \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} , \quad (7.4)$$

where the game  $\mathcal{G}_i$  captures the tolerated adversarial behaviour against the hybrid protocol  $\varrho_i(\pi_i)$ . As a result, the game chain converts a real world adversary  $\mathcal{A}$  step by step to an ideal world adversary  $\mathcal{A}^\circ$  such that

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon_1 + \dots + \varepsilon_s \quad (7.5)$$

and thus there exists a composite reduction schema

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon_s + \dots + \varepsilon_1]{***} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \quad (7.6)$$

provided that  $(\mathcal{G}_i, \mathcal{G}_{i-1}) \in \mathfrak{G}_i$  for all  $i \in \{1, \dots, s\}$  and all reduction steps can be carried out. As a result, security guarantees of all protocols  $\pi_1, \dots, \pi_s$  implicitly determine a set of valid game pairs  $\mathfrak{G}$ , where the compound protocol  $\pi$  remains  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -secure. For example, we often want to establish that the compound protocol  $\pi$  is secure in the stand-alone model.

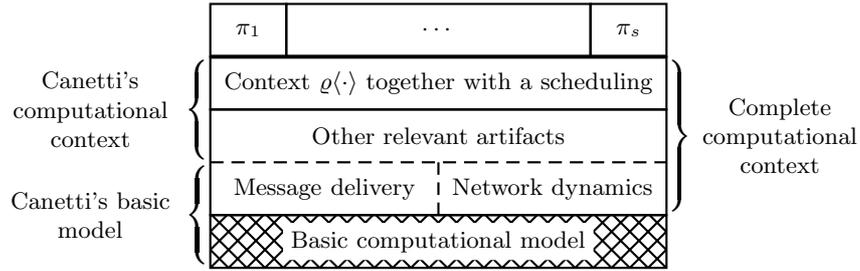


Figure 7.2: Layered description of a compound protocol  $\varrho\langle\pi_1, \dots, \pi_s\rangle$ .

Note that there are many trade-offs between efficiency and generality. Evidently, we obtain the most efficient solution when we tailor a protocol  $\pi$  specifically for a pre-specified context  $\varrho(\cdot)$ . However, such design cannot be reused and the corresponding security proof might be difficult to find. Alternatively, we may design a protocol  $\pi$  that is secure in almost any context. However, such a protocol might be impossible to devise or it might be highly inefficient. Thus, one has to find a proper balance between these extreme cases.

As a way out from this anarchy of security models, we require that each protocol is specified with usage restrictions that determine the supported contexts. Namely, one should fix a set of *composition rules* that determine the basic structural properties of plausible contexts; for example, whether the protocol shares some variables with the context or what kind of pre-, post- and side-computations are supported. Secondly, one should fix a tolerated adversarial behaviour and the maximal running time for computational contexts. Of course, any other unambiguous specification methodology is also appropriate.

## 7.2 LAYERED DESCRIPTION OF COMPUTATIONAL CONTEXTS

Although the general concept of computational context is quite intuitive, it is somewhat difficult to give a formalisation that is not plagued by the abundance of technical details. Various types of interactive computations share the same general description but differ in many minor details, for example compare the manuscripts [MR91b, Can00b, PSW00, BPW04]. To make technical details more tractable, we specify the computational model step by step. First, we define a basic model of computations that is absolutely minimal. Next, we formalise all other more complex artefacts, like asynchronous and non-authentic communication, in terms of this basic model. More precisely, we model these artefacts by adding extra nodes into the network and by adjusting the description of protocols and contexts. As a result, we can gradually formalise complex settings by grouping similar technical details together into separate layers, as illustrated in Fig. 7.2. Moreover, our task here is to specify only the basic computational model, since an appropriate formalisation of higher level artefacts is the responsibility of the person who analyses a concrete protocol or a context. Secondly, we can state and prove many theorems independently from higher level artefacts, as we can always analyse the basic level description of the protocol.

**Basic computational model.** The basic model consists of a fixed number of

participants  $\mathcal{P}_1, \dots, \mathcal{P}_m$  and a fixed number of trusted third parties  $\mathcal{T}_1, \dots, \mathcal{T}_s$  that have no inputs and no outputs. A trusted third party  $\mathcal{T}_i$  is active only in the context, where participants are supposed to execute an ideal implementation  $\pi_i^\circ$ . All participants of a context are modelled as interactive Random Access machines that run simultaneously at the same clock speed. Communication between participants is synchronous and secure. Moreover, a sender cannot proceed further until the message has been written on the recipients communication tape. Hence, the communication tape behaves as an unlimited buffer and the recipient does not have to process messages at the arrival time. The low-level message delivery procedure is a deterministic and a fairly low cost operation, which is described more thoroughly in Section 2.4.

**Adversaries and security games.** We treat adversaries as external entities and not as participants of a computational context. There are several important reasons for such a choice. First, our main aim here is to study attacks that utilise only acquired data and do not employ timings or other side information. Second, we formalised a computational context as a real-time system, where participants obey many real-time constraints. Hence, putting an adversary into the computational context would create many implicit limitations on the adversarial behaviour, which are normally neglected in cryptography. Third, timing-based attacks can be formalised also in our simplified model.

In our model, the adversary can influence computations by sending oracle calls to the challenger. Namely, the adversary can either corrupt participants, send some messages, or give his or her final output  $z_a$  directly to the challenger. A semi-honest corruption call provides read-only access to the internal state of the participant. A malicious corruption query gives a complete control over the participant  $\mathcal{P}_i$  and stops the local computations, i.e., the adversary must do all computations instead of  $\mathcal{P}_i$ . Additionally, we require that the adversary can send messages only on behalf of maliciously corrupted participants.

The challenger's primary task in the game is to correctly simulate the execution of a computational context and to serve the queries of an adversary. For the faithful execution, the challenger can allocate a separate thread for each participant and then simulate the execution in micro-rounds, where each thread completes only a single basic operation. Additionally, the challenger tests all queries and halts with  $\perp$  if the adversarial behaviour is not tolerable. As before, the security game itself is a ping-pong protocol started by the adversary. To avoid leakage of temporal information, the adversary is activated only when the challenger provides a reply or a corrupted participant receives a message.

**Computational complexity.** As a final detail, we must specify several important time bounds. Let  $t_\pi$  denote the time complexity of a protocol  $\pi$ , i.e., the number of the elementary steps needed to complete  $\pi$ . Similarly, let  $t_{\text{cnt}}$  denote the time complexity of a computational context  $\varrho(\cdot)$ , i.e., the number of elementary steps made by all non-adversarial parties in the game  $\mathcal{G}_{\text{ideal}}$ . The latter can be further decomposed into the online and the offline time complexity:

$$t_{\text{cnt}} = t_\varrho + t_{\text{pr}} \quad , \quad (7.7)$$

where  $t_\varrho$  counts all elementary steps made by  $\mathcal{P}_1, \dots, \mathcal{P}_m, \mathcal{T}_1, \dots, \mathcal{T}_s$  for evaluating  $\varrho(\pi^\circ)$  and  $t_{\text{pr}}$  is the time complexity of the corresponding security objective

$\mathcal{B}(\cdot)$ . As before, let  $t_{\text{re}}$  and  $t_{\text{id}}$  denote the running times of real and ideal world adversaries. Also, note that  $t_{\varrho}$  may depend on the adversarial behaviour.

**Message delivery.** Different algorithmic models of message delivery can be easily specified sending messages through a dedicated *courier* node  $\mathcal{P}_{\text{nw}}$ . The courier node  $\mathcal{P}_{\text{nw}}$  is an ordinary participant in the basic model, where communication is synchronous and secure. However, as all other participants send their messages via  $\mathcal{P}_{\text{nw}}$ , the effects of insecure, non-reliable, non-instantaneous and non-authentic communication can be modelled by an appropriate choice of the message delivery algorithm executed by  $\mathcal{P}_{\text{nw}}$ . For example, non-authentic communication can be formalised by using maliciously corrupted  $\mathcal{P}_{\text{nw}}$ . Similarly, a semi-honest corruption of  $\mathcal{P}_{\text{nw}}$  represents the ability to eavesdrop all communication channels without the possibility to alter messages.

Most importantly, the courier node  $\mathcal{P}_{\text{nw}}$  is an ordinary protocol participant and thus the accuracy and correctness of all these message delivery models is not our responsibility. Of course, the delivery model has to be specified, but such non-essential technical details do not clutter the basic formalism.

**Dynamic networks.** Dynamic evolution of computational contexts is another technical detail, which is emphasised a lot in Canetti’s manuscript [Can00b]. Again, such behaviour can be modelled by a dedicated *administrator* node  $\mathcal{P}_{\text{na}}$  that creates and deletes network nodes. More formally, only few nodes are initially active in the context and the remaining nodes are waiting for start-up messages from  $\mathcal{P}_{\text{na}}$ . Now any active node  $\mathcal{P}_i$  can send a special message with a program code to  $\mathcal{P}_{\text{na}}$  to activate a participant. The administrator  $\mathcal{P}_{\text{na}}$  loads the code into first free node  $\mathcal{P}_j$  and sends the *node label*  $j$  back to  $\mathcal{P}_i$ . Analogously, we can give nodes the ability to halt other nodes. Note that the bound on the total running time also limits the maximal number of activated nodes and thus  $\mathcal{P}_{\text{na}}$  never runs out of free nodes if the context has enough participants.

**Message scheduling.** Recall that a computational context  $\varrho\langle\pi\rangle$  specifies only a black-box usage of  $\pi$ . Such a specification is sufficient for the ideal implementation  $\pi^\circ$ , since the protocol execution is determined by a black-box usage pattern. However, normal protocols usually have a more complex structure and thus the message scheduling is not uniquely fixed. There are two principal ways to solve this ambiguity. First, we might fix the explicit scheduling for  $\varrho\langle\pi\rangle$  and consider the corresponding game pairs. However, such a choice is often unsuitable for practical applications, where messages are scheduled dynamically in order to minimise network delays. Similarly, a rigid scheduling is unsuitable for adaptive computations, where the execution itself depends dynamically on the non-deterministic choices made by the participants. As a way out, we allow flexible schedulings as long as they are completely deterministic. More formally, if we fix random coins of all participants and the adversary, then the message scheduling should be uniquely fixed for the protocol  $\varrho\langle\pi\rangle$ . We remind here that all artefacts are defined in the basic model and thus these random coins uniquely fix the networking behaviour and message delays.

**Other relevant artefacts.** Many other factors can influence the outcome of interactive computations starting from different clock speeds and ending with power consumption and sound made by the computational device. Nevertheless, all these artefacts can be implemented as dedicated nodes with a prescribed corruption level, or as extra messages in protocols. Nevertheless, the layered de-

scription of computational contexts is not a silver bullet. It just provides a more modular way to specify features and at the end of the day the overall complexity is comparable to Canetti’s formalism [Can00b]. In fact, Canetti’s model can be viewed as a canonical formalisation of interactive computations.

### 7.3 TWO FLAVOURS OF STAND-ALONE SECURITY

Although the stand-alone security model is a quite restrictive computational context, it is still a good starting point for illustrating many intricate concepts. In fact, simplicity is the main virtue of the stand-alone model, since the corresponding security proofs are quite straightforward and without subtle details. Hence, complex security notions have often alternative description in the stand-alone model to escape tedious technical details. In particular, it is important to characterise computational contexts that preserve stand-alone security.

First, note that participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  may possess more information than is actually used in the protocol. That is, let  $\mathbf{x} = (x_1, \dots, x_n)$  denote partial inputs used by the protocol and let  $\phi = (\phi_1, \dots, \phi_n)$  denote the corresponding complete inputs. We emphasise here that such a distinction is purely syntactic and not a redefinition of the stand-alone security model. Namely, we can always investigate protocols that first extract sub-inputs  $\mathbf{x}$  from the true inputs  $\phi$  and then use  $\mathbf{x}$  in later computations. Similarly, we use  $\psi = (\psi_1, \dots, \psi_n, \psi_a)$  to denote the final output states of all participants and the adversary.

Second, note that our and the classical formalisation of stand-alone security are qualitatively equivalent for static adversaries. Recall that a correspondence between the real and the ideal world adversaries  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  may depend on the security goal  $\mathcal{B}(\cdot)$  in our formalism, whereas the classical formalism requires a mapping  $\mathcal{A} \mapsto \mathcal{A}^\circ$  that is independent of  $\mathcal{B}(\cdot)$ , see Section 5.3. Now assume that  $\mathcal{P}_i$  is a non-corruptible node with an input  $\phi_i = (\beta_i, x_i)$  and let  $\psi_i = (\beta_i, x_i, y_i)$  be the corresponding output. Then we can consider a universal security goal  $\mathcal{B}_u^i(\cdot)$  that interprets  $\beta_i$  as a formal description of  $\mathcal{B}(\cdot)$  and outputs  $\mathcal{B}(\psi_1, \dots, \psi_n, \psi_a)$ . Due to the properties of RAM machines, there exists a constant  $c > 0$  such that any  $t_{\text{pr}}$ -time predicate  $\mathcal{B}(\cdot)$  can be interpreted in time  $c \cdot t_{\text{pr}}$ . Hence, if for any input distribution  $\mathfrak{D}$  and  $c \cdot t_{\text{pr}}$ -time predicate  $\mathcal{B}_u^i(\cdot)$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon, \quad (7.8)$$

then the mapping  $(\mathcal{B}_u^i, \mathcal{A}) \mapsto \mathcal{A}^\circ$  produces ideal world adversaries such that

$$\text{cd}_\star^{t_{\text{pr}}}(\mathcal{G}_{\text{re-atk}}^{\mathcal{A}}, \mathcal{G}_{\text{id-atk}}^{\mathcal{A}^\circ}) \leq \varepsilon \quad (7.9)$$

whenever  $\mathcal{P}_i$  remains uncorrupted. Consequently, there also exists a universal mapping  $\mathcal{A} \mapsto \mathcal{A}^\circ$  for all adversaries that corrupt a fixed set of nodes, as always assumed in the classical formalisations [Can00a, Gol04]. We emphasise that the equivalence does not hold for input-privacy in the malicious model.

Third, note that it is irrelevant whether a predicate  $\mathcal{B}(\cdot)$  is computed in centralised or distributed manner. Hence, stand-alone security is sufficient for all computational contexts  $\varrho(\cdot)$ , where all participants first execute a protocol and then interactively post-process the outputs  $\psi$ . However, the models are equivalent only if we can embed the post-processing phase into  $\mathcal{B}(\cdot)$ . For obvious

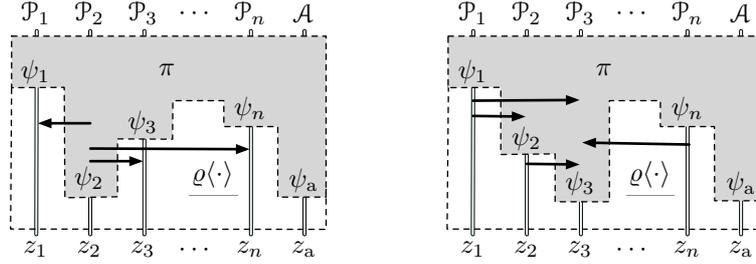


Figure 7.3: Synchronisation errors that break the equivalence with the stand-alone setting. Escaping messages are on the left and invading messages on the right. The execution time goes from top to bottom.

reasons, such an embedding does not exist for input-privacy but there are other more surprising exceptions. Namely, the adversarial behaviour might cause synchronisation errors, where protocol messages are mixed together with the post-processing phase, see Fig. 7.3. The grey area in the figure represents the time when the participants execute the protocol, the white area represents the post-processing phase and the arrows represent misinterpreted messages.

In short, messages can either escape or invade the stand-alone setting. A message *escapes* from the stand-alone setting if it is sent to a participant that has already completed the protocol. A message *invades* the stand-alone setting if it is sent by a node that has already completed the protocol. To avoid ambiguity, we say that all maliciously corrupted nodes finish the protocol together with the last (semi-)honest node. Escaping messages have no influence on outputs  $\psi$  in the stand-alone model, since the low level message delivery just writes them on the communication tapes of halted participants. Therefore, we cannot construct the corresponding predicate, since the post-processing phase may actually use these messages. Invading messages cause a similar incompatibility with the stand-alone model. Moreover, it is straightforward to construct examples, where such synchronisation errors cause the complete failure of a protocol that is secure in the stand-alone model. We leave this as an exercise to the reader.

We emphasise here that it is relatively easy to eliminate problems caused by escaping messages. For example, we can use special tags to denote the protocol messages and drop all these messages in the post-processing stage. Invading messages, on the other hand, cause more severe problems, since they bring extra information to the stand-alone setting. In particular, a post-processing context  $\varrho(\cdot)$  may force a participant  $\mathcal{P}_i$  to publish the state  $\psi_i$  after completion of the protocol and such synchronisation errors are extremely dangerous.

As an illustrative example consider a lottery protocol  $\pi_{sl}$  depicted in Fig. 7.4, where  $\mathcal{P}_2$  tries to guess a random number  $s_1$  generated by  $\mathcal{P}_1$  and the number  $s_1$  is sent to the arbiter  $\mathcal{P}_3$  to prevent  $\mathcal{P}_1$  from cheating. In the ideal implementation, the trusted third party does all computations by herself and sends the results back to the participants. Now assume that only  $\mathcal{P}_2$  can be corrupted and that all communication channels are secure. Then the protocol is  $2^{-k+1}$ -secure in the malicious model. The corresponding simulator  $\text{Sim}$  just ignores the messages sent by  $\mathcal{T}$  and  $\mathcal{P}_2$  and outputs whatever  $\mathcal{A}$  does. However, the protocol  $\pi_{sl}$  is clearly insecure in the post-processing context, where  $\mathcal{P}_2$  queries  $s_1$  from  $\mathcal{P}_3$ .

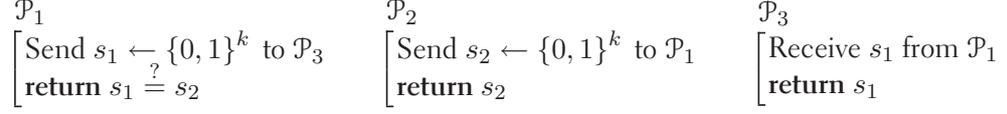


Figure 7.4: Protocol  $\pi_{\text{sl}}$  that models a simple lottery with an arbiter  $\mathcal{P}_3$ .

Namely,  $\mathcal{P}_2$  can first pretend that  $\pi_{\text{sl}}$  is completed to get the *invading* message  $s_1$  from  $\mathcal{P}_3$  and then forward  $s_1$  to  $\mathcal{P}_1$  and thus always win the lottery. At the same time, the ideal implementation remains secure in this context. Moreover, the underlying problem cannot be solved with message tagging.

Such protocol failures are extremely dangerous and should be avoided by design. A protocol  $\pi$  has a *robust message scheduling* if no tolerable attack can force (semi-)honest nodes to make synchronisation errors. In particular, honest participants can always detect when all other honest participants have finished the protocol. A robust scheduling is always achievable in the presence of an authentic broadcast channel, as each participant can broadcast a specific message when he or she completes the protocol. Of course, there are other alternatives, such as fixed round protocols or limits on the duration of  $\pi$ .

**Strong stand-alone security model.** The example above clearly shows that a protocol  $\pi$  must have a robust scheduling, or otherwise it might lose security in post-processing contexts. However, the latter is not always sufficient when a context has a pre-processing phase. In particular, note that the adversary might gain some knowledge  $\phi_a$  in the pre-computation phase that helps to attack the protocol afterwards. In the following, we describe a *strong stand-alone security model* that together with robust messaging assures security in all contexts  $\varrho(\cdot)$ , where the protocol  $\pi$  is executed without any side-computations.

The ideal attack phase  $\mathcal{G}_{\text{id-atk}}$  coincides with the standard formalisation described in Section 5.3 but there is a minor change in the real attack phase  $\mathcal{G}_{\text{re-atk}}$ . Namely, the challenger must always notify the adversary when some participant halts. Secondly, we consider the extended inputs  $\phi_e = (\phi_1, \dots, \phi_n, \phi_a)$  instead of normal inputs  $\phi = (\phi_1, \dots, \phi_n)$  but otherwise there are no other differences with Section 5.3. In particular, a real and an ideal world model are  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t. a set of relevant predicates  $\mathfrak{B}$  in the strong stand-alone model if for any security objective  $\mathcal{B}(\cdot) \in \mathfrak{B}$  and for any  $t_{\text{re}}$ -time real world adversary  $\mathcal{A}$  there exists a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any extended input distribution  $\phi_e \leftarrow \mathfrak{D}_e$ :

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (7.10)$$

As a result, we get a slightly stronger reduction schema

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon]{\text{MPC-SEC}^+} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} . \quad (7.11)$$

Note that we can always pack an extended distribution  $\mathfrak{D}_e$  into a standard input distribution  $\mathfrak{D}$  by treating the input of  $\mathcal{P}_i$  as a pair  $(\phi'_i, \phi'_a)$  and the other way around. As a result, it is straightforward to show that the additional knowledge  $\phi_a$  obtained in the pre-computation stage does not increase the adversary's ability to attack a protocol in the case of static corruption. Consequently, the extra input

$\phi_a$  is essential only if an adaptive or mobile adversary uses it to find “optimal” candidates for further corruption, see [Can00a, p.179].

Secondly, we can make a mapping  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  algorithmic for any fixed security goal  $\mathcal{B}(\cdot)$ . Let  $\mathcal{A}_u$  be a universal adversary that interprets  $\phi_a$  as a formal description of an adversarial strategy  $\mathcal{A}$ . Then the corresponding ideal adversary  $\mathcal{A}_u^\circ$  can be viewed as a compiler that devises an appropriate attack strategy for the code  $\phi_a$ . However, the algorithmic correspondence is not for free, since the adversaries  $\mathcal{A}_u^\circ(\phi_a)$  and  $\mathcal{A}^\circ$  have different running times  $t_{\text{id}}(c \cdot t_{\text{re}})$  and  $t_{\text{id}}(t_{\text{re}})$  where the constant  $c > 0$  is the interpreting overhead.

For static adversaries, we can combine two results by considering the universal security goal  $\mathcal{B}_u^i(\cdot)$  and the universal adversary  $\mathcal{A}_u$ . The corresponding compiler  $\mathcal{A}_u^i(\cdot)$  assures also the computational indistinguishability of outputs when node  $\mathcal{P}_i$  is not corrupted. Hence, our and classical formulation of stand-alone security coincide even if the set of corrupted parties is randomly fixed by the adversary. For the proof, consider a super-compiler  $\mathcal{A}_u^*(\cdot)$  that first fixes random coins  $\omega$  for the adversary  $\mathcal{A}$  and obtains the list of corrupted participants  $\mathcal{C}$ . Next,  $\mathcal{A}_u^*(\cdot)$  chooses  $i$  such that  $\mathcal{P}_i \notin \mathcal{C}$  and uses the compiler  $\mathcal{A}_u^i(\cdot)$  to transform the deterministic algorithm  $\mathcal{A}(\omega)$  and executes the end result.

Finally, we want to emphasise that the existence of efficient compiler constructions is not formally sufficient for the constructive proofs needed for subjective security. Although such claim seems to be contradictory at first sight, it has a straightforward explanation. The classical security proof might prove non-constructively that the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  exists. The latter proves also the existence of the universal compiler  $\mathcal{A}_u^\circ(\cdot)$  but does not provide an explicit code for it. Consequently, the subjective security proof is still incomplete as we cannot *explicitly* write down the code of a universal compiler  $\mathcal{A}_u^\circ(\cdot)$ . The latter is not a mere theoretical discussion, as some cryptographic proofs actually use inefficient counting arguments, e.g. [BL07].

Also, the existence of such compilers  $\mathcal{A}_u^\circ(\cdot)$  does not contradict classical impossibility results for program analysis, since the compiler handles only programs with a fixed size. As a result, the compiler  $\mathcal{A}_u^\circ(\cdot)$  may use an ultra-compact advice string—Philosopher’s Stone—to speed up the online code analysis phase. Buldas and Laur [BL07] showed that such a Philosopher’s Stone must exist for a very specific problem, but the corresponding counting argument itself is quite universal. In a nutshell, white-box proofs exist even for cryptographic primitives that satisfy only abstract security properties.

## 7.4 CANONICAL DECOMPOSITION TECHNIQUES

High-level security proofs often split interactive computations into isolated sub-phases to simplify the security analysis. For example, stand-alone security guarantees are sufficient for all interactive post-processing phases that can be embedded into the predicate  $\mathcal{B}(\cdot)$ . Although this claim seems trivial, the corresponding formal proof must separate post-processing phase from the protocol execution and formalise it as a sub-formula in the predicate  $\mathcal{B}(\cdot)$ . However, the robust scheduling only assures that the computational process is logically separable, i.e., messages are always sent and received in the same computational phase and we can view internal states  $\psi = (\psi_1, \dots, \psi_m, \psi_a)$  at the end of the phase as

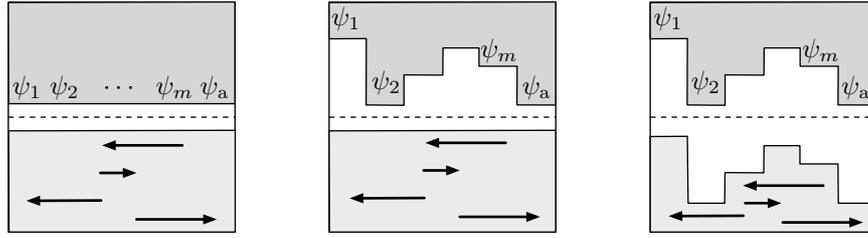


Figure 7.5: Three basic ways to decompose interactive computations into two sub-phases without altering the final output vector  $\mathbf{z}_{\text{obs}}$ .

inputs  $\phi = (\phi_1, \dots, \phi_m, \phi_a)$  in the next phase. However, different participants can finish the same phase in different time moments, as depicted in Fig. 7.5. Therefore, the knowledge of the internal states  $\psi$  alone might be insufficient to reconstruct the computations in the following phase. Only a slight change in timings may influence the ordering of messages and thus have an effect on the final outputs  $\mathbf{z}_{\text{obs}}$  even if the protocol and the context are specified in terms of received messages and their order, and not on their arrival times.

There are three principal solutions for this problem. First, we can guarantee that all participants always finish sub-phases simultaneously and thus the timing issues become irrelevant. In the corresponding synchronised communication model, messages are sent in only during short rounds that occur in fixed time intervals, and the exact order of messages in a round is discarded. Such a simple model for message delivery is used in many classical works, see for example [GMW87, BOGW88, CCD88, Bea91b, MR91b, DM00, Can00a].

As a second alternative, we can include timing effects directly to our computational model. More formally, we assume that the challenger is equipped with a clock<sup>1</sup> that is set to zero at the beginning of computations. In this model, observable outputs are pairs  $(z_i, \tau_i)$ , where  $z_i$  is a classical output and  $\tau_i$  is the exact halting time. As a result, we can talk about time-preserving correspondences between real and ideal world models, since a security goal  $\mathcal{B}(\cdot)$  now depends on exact halting times  $\tau_1, \dots, \tau_m$ . Naturally, such a correspondence is possible only if the adversary has some control over timings. In the corresponding formal model, the challenger always adds timing information to his or her replies and the adversary can specify the exact time when the challenger must carry out the corresponding oracle queries. However, the corresponding *real-time model of computations* is quite technical and takes us away from our main goal—the study of time-invariant protocols and contexts. Hence, we do not discuss this model further, although the following proofs can be directly generalised to handle exact timing in computations.

A third alternative is to eliminate all timing artefacts from the model so that knowledge of internal states  $\psi$  alone is sufficient to reconstruct the proceeding computations. For example, we can postulate that the order of messages is independent of the inputs and the adversarial behaviour. Then the knowledge of  $\psi$  uniquely determines the output, since the order of messages is fixed but unknown. However, adversarial forces can often influence message delivery and

<sup>1</sup>The clock models a global physical time in the actual computations.

thus such models with fixed message schedulings are overly optimistic. Evidently, we can route messages through special courier nodes  $\mathcal{P}_{\text{nw}}^1, \dots, \mathcal{P}_{\text{nw}}^v$  in order to model network delays. However, slight changes in timings can change the order of messages received by courier nodes  $\mathcal{P}_{\text{nw}}^u$  and the correct decomposition is still impossible without the exact timing information  $\tau$ .

In more formal terms, a computational process with several parallel threads is often time-sensitive even if the individual threads are time-invariant. Therefore, we cannot escape timing artefacts, unless we consider pseudo-parallel executions, where only a single participant is active at the same time moment. Let  $\mathcal{P}_{\text{sc}}$  be a special malicious *scheduler node* that can activate other participants in any order as long as no two nodes are simultaneously active. That is, all other participants are initially waiting for a special activation messages from  $\mathcal{P}_{\text{sc}}$ . When a semi-honest node receives an activation message, the node does some computations and gives control back to  $\mathcal{P}_{\text{sc}}$ . If the activated node is malicious, the control goes back to the adversary. As a result, the adversary has full control over the scheduling and the knowledge of internal states  $\psi$  is sufficient for correct decompositions as desired. The corresponding *asynchronous model of computations* has many variations, since there are many reasonable alternatives for scheduling and network behaviour. Still, some formalisations are more fundamental than the others. In particular, all classical computational models [BOCG93, Can00b] require that an activated node stops when it starts to read the next incoming message. Such a scheduling has optimal granularity, since the adversary can dynamically choose any valid message ordering, but the scheduling is coarse enough to hide exact timing.

## 7.5 CHARACTERISATION OF SEQUENTIAL COMPOSABILITY

In the following section, we study the limitations of strong stand-alone security. In particular, we formally prove that strong stand-alone security is sufficient for all computational contexts, where honest participants stop all other computations during the protocol. Although such an *isolated execution* policy is quite restrictive, it is still adequate for many practical settings. In a sense, this result formally validates the strong stand-alone security model, as it shows that the model provides practically meaningful security guarantees.

For clarity, we divide the corresponding security analysis into two phases. First, we formally prove that any post-processing phase can be embedded into the security goal  $\mathcal{B}(\cdot)$  if a protocol has a robust message scheduling. Second, we show that the effects of pre-computation phase can always be modelled as an extended input distribution  $\mathcal{D}_e$ . As a result, the security of an isolated protocol execution follows from the strong stand-alone security. In particular, a compound protocol that executes sub-protocols one by one remains secure in the strong stand-alone model as long as sub-protocols are also secure in the strong stand-alone model, i.e., strong stand-alone security is *sequentially composable*. We emphasise here that these claims hold only for the computational models that support *errorless decompositions*, i.e., the knowledge of internal state  $\psi$  is sufficient to determine the final outputs  $z_{\text{obs}}$ .

**Lemma 1.** *If a protocol has a robust message scheduling, then any post-processing phase can be embedded into the security goal. The corresponding overhead*

is linear in the total complexity of the post-processing phase.

*Proof sketch.* For the proof, we split a pair  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  of corresponding security games into two sub-phases. In the first phase, (semi-) honest participants execute the original context until the real protocol  $\pi$  or the ideal protocol  $\pi^\circ$  is completed and output their internal states  $\psi_i$ . After all participants have finished, the corresponding output vector  $\boldsymbol{\psi} = (\psi_1, \dots, \psi_m, \psi_a)$  is used as an input vector for the second phase, where (semi-)honest participants restore their internal states and continue with the execution. The vector of final outputs  $\mathbf{z}_{\text{obs}}$  is used as an input to the original security goal  $\mathcal{B}(\cdot)$ . Let us denote the corresponding two-phase security games by  $\mathcal{G}_{\text{real}}^*$ ,  $\mathcal{G}_{\text{ideal}}^*$  and the corresponding sub-adversaries by  $\mathcal{A}_1, \mathcal{A}_2$  and  $\mathcal{A}_1^\circ, \mathcal{A}_2^\circ$ . Next, we define constructive mappings  $\mathcal{A} \mapsto (\mathcal{A}_1, \mathcal{A}_2)$  and  $(\mathcal{A}_1^\circ, \mathcal{A}_2^\circ) \mapsto \mathcal{A}^\circ$  that achieve perfect correspondence

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \equiv (\mathcal{G}_{\text{real}}^*)^{\mathcal{A}_1, \mathcal{A}_2} \quad \text{and} \quad (\mathcal{G}_{\text{ideal}}^*)^{\mathcal{A}_1^\circ, \mathcal{A}_2^\circ} \equiv \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ}. \quad (7.12)$$

More precisely, if we additionally assume that the challenger notifies  $\mathcal{A}_1$  in the game  $\mathcal{G}_{\text{real}}^*$  when a (semi-)honest participant halts, then the correspondence  $\mathcal{A} \mapsto (\mathcal{A}_1, \mathcal{A}_2)$  is evident. Namely, the sub-adversary  $\mathcal{A}_1$  can run  $\mathcal{A}$  until all (semi-)honest nodes are halted<sup>2</sup> and output the corresponding internal state as  $\psi_a$ . In the second phase, the sub-adversary  $\mathcal{A}_2$  restores the internal state of  $\mathcal{A}$  and continues the execution of  $\mathcal{A}$ . For the second mapping  $(\mathcal{A}_1^\circ, \mathcal{A}_2^\circ) \mapsto \mathcal{A}^\circ$ , note that the adversary  $\mathcal{A}^\circ$  can always detect the change of phases in the ideal world and thus the adversary  $\mathcal{A}^\circ$  can sequentially run  $\mathcal{A}_1^\circ$  and  $\mathcal{A}_2^\circ$ .

Now it is straightforward to prove that all decompositions are well defined and the equivalences (7.12) indeed hold if  $\pi$  has a robust scheduling and the computational model facilitates errorless decomposition. Also, the difference in running times is linear. Thus, we can analyse game pairs  $(\mathcal{G}_{\text{real}}^*, \mathcal{G}_{\text{ideal}}^*)$  and mappings  $(\mathcal{A}_1, \mathcal{A}_2) \mapsto (\mathcal{A}_1^\circ, \mathcal{A}_2^\circ)$  instead of  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  and  $\mathcal{A} \mapsto \mathcal{A}^\circ$ . As the real and ideal world are identical after the first phase, we can always choose  $\mathcal{A}_2^\circ = \mathcal{A}_2$ . Consequently, the second phase together with  $\mathcal{B}(\cdot)$  can be viewed as a new security objective  $\mathcal{B}'(\cdot)$  and the claim follows.  $\square$

**Lemma 2.** *Any pre-processing phase with robust scheduling can be replaced with an extended input distribution  $\mathfrak{D}_e$ . The corresponding overhead in running times is linear for the context and for the adversary.*

*Proof sketch.* Again, we can split a pair  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  of corresponding security games into two sub-phases so that the first phase captures the pre-processing phase and the second phase captures the succeeding computations. Let us denote the corresponding two-phase game pair by  $(\mathcal{G}_{\text{real}}^*, \mathcal{G}_{\text{ideal}}^*)$  and the vector of internal states between the sub-phases by  $\boldsymbol{\phi} = (\phi_1, \dots, \phi_m, \phi_a)$ . Similarly, to the proof of Lemma 1, we can analyse game pairs  $(\mathcal{G}_{\text{real}}^*, \mathcal{G}_{\text{ideal}}^*)$  and mappings  $(\mathcal{A}_1, \mathcal{A}_2) \mapsto (\mathcal{A}_1^\circ, \mathcal{A}_2^\circ)$  instead of  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  and  $\mathcal{A} \mapsto \mathcal{A}^\circ$ . We omit the corresponding argumentation, since it is completely analogous to the previous proof. Finally, note that the first phase can be replaced with sampling  $\boldsymbol{\phi}_e \leftarrow \mathfrak{D}_e$  in the games  $\mathcal{G}_{\text{real}}^*$  and  $\mathcal{G}_{\text{ideal}}^*$ , if we consider only mappings with  $\mathcal{A}_1^\circ = \mathcal{A}_1$ . Again such a choice is natural, since the ideal and real world coincide in the first phase.  $\square$

<sup>2</sup>The escaping messages produced by the first stage of the adversary should be re-sent in the second stage to assure equivalence.

These decomposition lemmas alone are sufficient to prove several sequential composability results, provided that the computational context  $\varrho(\cdot)$  and the protocol  $\pi$  share the same set of participants. However, this assumption is rarely valid in practical applications, where only a few participants  $\mathcal{P}_{i_1}, \dots, \mathcal{P}_{i_n}$  of the global context  $\varrho(\cdot)$  take part in the protocol  $\pi$ . In fact, participants  $\mathcal{P}_{i_1}, \dots, \mathcal{P}_{i_n}$  usually have little, if any, knowledge about the global context  $\varrho(\cdot)$ . Also, it is unlikely that participants  $\mathcal{P}_{i_1}, \dots, \mathcal{P}_{i_n}$  can force some global restrictions on the other participants. However, local restrictions concerning only the participants of the protocol  $\pi$  are enforceable. In particular, participants can always run the protocol  $\pi$  in *isolation* by forcing the following local restrictions:

- (S1) The context  $\varrho(\cdot)$  uses the protocol  $\pi$  in black-box manner.
- (S2) The protocol  $\pi$  and pre-processing phase have robust message scheduling when we consider only the participants of the protocol  $\pi$ .
- (S3) The participants of the protocol  $\pi$  send out only the messages of  $\pi$  and delete all non-protocol messages during the execution of the protocol  $\pi$ .

Note that there are several ways how the participants of the protocol  $\pi$  can relate to other participants. If the protocol  $\pi$  provides an end-to-end solution, then it is also reasonable to assume that all non-participants are corrupted, as these external entities are out of our control. For example, all those millions and millions of computational devices that are connected to Internet, but do not participate in our protocol, are potentially hostile non-participants that we cannot control.

In the two-party setting, it is natural to assume that non-participants are corrupted, since each party trusts only him- or herself. The same is true for multi-party protocols, where one controls a fixed set of nodes and thus the adversary can potentially abuse all signals that are sent out of this group. On the other hand, the group of semi-controllable nodes that follow the restrictions (S1)–(S3) might change in time. For example, some new members that are believed to be honest can join the group. Therefore, it makes sense to slightly relax the assumption and postulate the following:

- (A1) All nodes that do not participate in the protocol  $\pi$  are compromised during the execution of  $\pi$ , afterwards the adversary might retreat from them.
- (A2) All trusted third parties that are guaranteed to finish before or start after the execution of  $\pi$  can be considered as participants of the protocol  $\pi$ .

Now it is straightforward to establish a correspondence between the isolated protocol execution and the strong stand-alone security model, i.e., the strong stand-alone security is sufficient for all security levels except for input-privacy. For clarity, we state only the qualitative results. The exact quantitative relations between the time bounds can be extracted from the proofs if necessary.

**Theorem 1.** *A context  $\varrho(\cdot)$  and a protocol  $\pi$  that satisfy restrictions (S1)–(S3) and (A1)–(A2) preserve strong stand-alone security. The corresponding overhead is linear for the running times and the bound on the advantage does not change.*

*Proof sketch.* Let us first consider the simplified case, where the participants of the context  $\varrho(\cdot)$  and the protocol  $\pi$  coincide. Since the assumptions of Lemmata 1 and 2 are satisfied, we can split the respective game pair  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  into

three sub-phases, where the first phase corresponds to the pre-processing phase and the third one to the post-processing phase. Moreover, there are constructive mappings  $\mathcal{A} \mapsto (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$  and  $(\mathcal{A}_1^\circ, \mathcal{A}_2^\circ, \mathcal{A}_3^\circ) \mapsto \mathcal{A}^\circ$  such that

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \equiv (\mathcal{G}_{\text{real}}^*)^{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3} \quad \text{and} \quad (\mathcal{G}_{\text{ideal}}^*)^{\mathcal{A}_1^\circ, \mathcal{A}_2^\circ, \mathcal{A}_3^\circ} \equiv \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} . \quad (7.13)$$

To conclude the proof, we must construct a mapping  $\mathcal{A}_2 \mapsto \mathcal{A}_2^\circ$  such that

$$|\Pr [(\mathcal{G}_{\text{real}}^*)^{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3} = 1] - \Pr [(\mathcal{G}_{\text{ideal}}^*)^{\mathcal{A}_1, \mathcal{A}_2^\circ, \mathcal{A}_3} = 1]| \leq \varepsilon . \quad (7.14)$$

Lemma 1 assures that the third phase can be embedded into the security goal with a linear overhead. Lemma 2 assures that the first phase can be replaced with an appropriate extended distribution  $\mathfrak{D}_e$ . As the second phase corresponds to the strong stand-alone security model, the existence of a mapping  $\mathcal{A}_2 \mapsto \mathcal{A}_2^\circ$  is implied by the assumptions of the theorem. The claim holds even if the exact roles in the protocol are determined dynamically by the context, since we can always assume that the internal states  $\phi$  and  $\psi$  also contain the formal specification (code) of the computational context.

For the general case, consider a new extended protocol  $\hat{\pi}$ , where the non-participants of the protocol  $\pi$  just output their initial states and halt. Then the original security game  $\mathcal{G}_{\text{real}}$  can be viewed as an execution of  $\hat{\pi}$ , where the adversary corrupts the non-participants during the execution to do some malicious side-computations. Note that the protocol  $\hat{\pi}$  and the pre-processing phase formally have a robust scheduling, since all trusted third parties are guaranteed to be inactive and other non-participants are assumed to be corrupted during the execution of the protocol  $\hat{\pi}$ . Thus, we have reached the simplified case.

To conclude, note that the modified protocol  $\hat{\pi}$  preserves strong stand-alone security possibly with linear overhead. First, the presence of inactive trusted third parties is irrelevant, since their inputs and outputs are empty. Second, there is one-to-one correspondence between the security games for the protocols  $\pi$  and  $\hat{\pi}$ . Namely, the inputs of other non-participants can be embedded into the adversarial input  $\phi_a$  and the outputs can be embedded into the output  $\psi_a$ . Similarly, any security goal  $\mathcal{B}(\cdot)$  for the protocol  $\hat{\pi}$  can be translated to a predicate  $\mathcal{B}'(\cdot)$  that first extracts all embedded outputs from  $\psi_a$  and then evaluates  $\mathcal{B}(\cdot)$ . Hence, it is straightforward to construct an appropriate mapping  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  for the protocol  $\hat{\pi}$  given only a mapping  $(\mathcal{B}', \mathcal{A}) \mapsto \mathcal{A}^\circ$  for the protocol  $\pi$ . The corresponding overhead is linear, as needed.  $\square$

**Corollary 1.** *Security in the strong stand-alone security model is sequentially composable, provided that all sub-protocols have robust scheduling.*

*Proof.* Let a protocol  $\pi$  be a sequential application of protocols  $\pi_1, \dots, \pi_s$  that share the same set of participants. Then Theorem 1 assures that we can substitute all protocols  $\pi_i$  step by step with the corresponding ideal implementations  $\pi_i^\circ$ . Although each substitution step introduces a new trusted third party, the assumption (A2) still holds and we can continue. The corresponding game chain completes the proof and provides necessary bounds on the advantage.  $\square$

**Further discussion.** The correspondence between the stand-alone security and the isolated protocol execution implies that the strong stand-alone security is a

sufficient and necessary security objective for end-to-end solutions. Also, note that protocol designers control only a tiny part of the whole world and the distinction between internal and external entities is unavoidable. The assumption (A1) captures the obvious judgement that external entities do not have to follow any restrictions. In that light, Theorem 1 is just a description that explains how and what kind of protocols one should execute in unknown hostile environments. We emphasise that the global computational context itself can reveal much more information than the ideal implementation of the protocol. Nevertheless, we cannot relax security requirements further to optimise the protocol performance, since we do not know how much extra information is revealed by the computational context. Consequently, the equivalence between ideal and real world implementations is the best we can achieve.

Historically, cryptographers have always taken the stand-alone security model for granted and have seldom explored its limitations in practical applications. The latter is the main reason why Theorem 1 or its analogues have never been proved in the mainstream cryptographic literature, although the result has been implicitly known for decades. In that sense, various composability results that simplify the design and analysis of protocols are more important. Thus, classical treatments of secure computations, such as [Can00a, Gol04], emphasise only sequential composability results and the corresponding design methodology.

Also, note that the equivalence result holds only if we assume that all external parties are corrupted. This assumption is often too conservative and can significantly weaken the corresponding security guarantees. The latter is another reason why the equivalence result has remained unnoticed, as other more stringent security notions provide more precise security guarantees.

Secondly, note that the usage restrictions (S1)–(S3) and (A1)–(A2) are optimal for the strong stand-alone security. In particular, stand-alone security is generally insufficient to guarantee parallel composability, see [GK96b, Lin04]. Hence, the assumption (A1) is essential and cannot be relaxed, since the restrictions (S1)–(S3) do not eliminate the possibility of parallel execution. For example, consider a context where  $\mathcal{P}_1, \mathcal{P}_2$  execute a protocol  $\pi_1$  and  $\mathcal{P}_3, \mathcal{P}_4$  execute a protocol  $\pi_2$  at the same time. Then the restrictions (S1)–(S3) and (A2) are formally satisfied, although the real implementation cannot be replaced with the ideal protocol. A concrete counterexample is given in [GK96b] as Theorem 5.1. The same counterexample can be modified to show that all trusted third parties must be inactive during the execution of the protocol.

Although the usage restrictions are optimal, we can still do some cosmetic relaxations. In particular, we can bypass the restriction (A1) by artificially increasing the number of participants. Let  $\hat{\pi}$  be an extended protocol where all non-participants are inactive during the execution of  $\pi$ . Then it is straightforward to prove that the protocol  $\hat{\pi}$  preserves strong stand-alone security if and only if the correspondence  $(\mathcal{B}, \mathcal{A}) \mapsto \mathcal{A}^\circ$  is independent of the security goal  $\mathcal{B}$ . Consequently, if the protocol  $\pi$  is secure according to classical definitions, there is no need to corrupt honest participants that remain inactive during the execution of  $\pi$ . However, this relaxation holds only for those honest participants that are guaranteed to be inactive, since the pre-processing phase and the protocol  $\hat{\pi}$  must have a robust message scheduling.

**Centralised task scheduling.** Note that strong stand-alone security is sufficient for modular protocol design, since sub-tasks can always be executed one by one.

As an advantage, we do not have to strengthen the security requirements for sub-protocols but there are also many important drawbacks.

Firstly, we must implement a global synchronisation service that reserves time slots for individual sub-tasks. As a result, all nodes must either participate in a protocol or wait until the protocol is completed. Thus, the whole network acts as a single virtual processor and load balancing is impossible. Moreover, the overall performance is determined by the slowest node in the network.

Secondly, note that modern computing devices do millions of operations in few milliseconds. Consequently, network delays can cause significant performance penalties. Moreover, the relative impact of network delays only increases in the future, as these delays are likely to remain constant, whereas the computational throughput is likely to grow in the future.

Thirdly, secure function evaluation is not the only practical design goal. In many cases, we need to implement secure services that have several rounds or are non-terminating. For example, access control and secure storage systems must function continuously. Centralised task scheduling makes it impossible to run two or more continuous services in parallel. Instead, we must divide the corresponding ideal functionalities into micro-rounds and use some kind of fair interleaving mechanism to assure the availability of all services. However, such a solution is artificial and inherently decreases the responsiveness.

## 7.6 CHARACTERISATION OF UNIVERSAL COMPOSABILITY

In many cases, it is impossible to guarantee that sub-protocols are not executed simultaneously or the corresponding performance penalty is prohibitively large. Hence, it is advantageous to consider more liberal task schedulings. The latter again requires more strict security definitions, since the stand-alone security does not assure the security of parallel executions [GK96b, Lin04]. Evidently, various restrictions on computational contexts lead to different security requirements and feasibility results, for example, compare the results of [Pas04] and [Lin04]. However, the fundamental limitations obtained in [Lin03a, Lin04] indicate that additional usage restrictions do not make it easier to design protocols, unless these restrictions are really specific. Hence, it makes sense to consider protocols that preserve the security in all time-bounded computational contexts.

Formally, we must first fix all lower level artefacts, such as message delivery and timing models. In particular, we can study synchronised, asynchronous or real-time model of computations. As usual let  $\pi$  be a protocol and  $\pi^\circ$  denote the corresponding ideal implementation. Let  $\mathfrak{G}_u$  be the set of all game pairs  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}})$  that are induced by all computational contexts  $\varrho(\cdot)$  and by all security objectives  $\mathcal{B}(\cdot)$ . Finally, fix an adversarial structure  $\mathcal{A}_{\text{loc}}$  that specifies the tolerable adversarial behaviour in local terms. The latter is necessary as contexts may assign different roles in the protocol  $\pi$  to different participants. Now the protocol  $\pi$  is  $(t_{\text{re}}, t_{\text{pr}}, t_{\text{cnt}}, \varepsilon)$ -universally composable w.r.t. the adversarial structure  $\mathcal{A}_{\text{loc}}$  if for any  $t_{\text{cnt}}$ -time game pair  $(\mathcal{G}_{\text{ideal}}, \mathcal{G}_{\text{real}}) \in \mathfrak{G}_u$  and for any  $t_{\text{re}}$ -time real world adversary  $\mathcal{A}$  that respects  $\mathcal{A}_{\text{loc}}$ , there exist a  $t_{\text{id}}$ -time ideal world adversary  $\mathcal{A}^\circ$  such that for any extended input distribution  $\mathfrak{D}_e$ , we have

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} = 1]| \leq \varepsilon . \quad (7.15)$$

The corresponding reduction schema is denoted as

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \xrightarrow[\varepsilon]{\text{UC-SEC}} \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} . \quad (7.16)$$

We remark that the original definition of universal composability [Can01] was given in different technical terms. The computational context  $\varrho\langle\cdot\rangle$  is substituted with an autonomous malicious entity  $\mathcal{Z}$  that has black-box access to the protocol  $\pi$  and interacts with the adversary. However, the difference is purely syntactical, as the environment  $\mathcal{Z}$  can always simulate the computational context  $\varrho\langle\cdot\rangle$  and vice versa, see also [Lin03b]. We prefer the notion of computational context, since it is conceptually simpler and closer to reality.

Also, note that our definition is based on a mapping  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}}, \mathcal{A}) \mapsto \mathcal{A}^\circ$ , whereas the original definition [Can01] requires universality  $\mathcal{A} \mapsto \mathcal{A}^\circ$  for all  $t_{\text{cnt}}$ -time game pairs  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}}) \in \mathfrak{G}_{\text{u}}$ . As a result, the original definition also violates the external consistency restrictions (C1)–(C2), whereas our definition is externally consistent. Still, these definitions are qualitatively equivalent. More formally, consider a universal context  $\varrho_{\text{u}}\langle\cdot\rangle$ , where all participants interpret their inputs as a code. Then the corresponding ideal world adversary  $\mathcal{A}_{\text{u}}^\circ$  is universal for all game pairs  $(\mathcal{G}_{\text{real}}, \mathcal{G}_{\text{ideal}}) \in \mathfrak{G}_{\text{u}}$  and the quantitative changes in the time bounds are linear. However, this equivalence result was first rejected in the asymptotic security model, since the initial definition of contexts with polynomial running times<sup>3</sup> was too strict, see [Can00b, p. 49–50].

Finally, note that universal composability is closed under concurrent composition. More formally, if a compound protocol  $\varrho_0\langle\pi_1^\circ\rangle$  is a universally composable implementation of a functionality  $\pi_0^\circ$  and a protocol  $\pi_1$  is a universally composable implementation of a functionality  $\pi_1^\circ$ , then the protocol  $\varrho_0\langle\pi_1\rangle$  is also a universally composable implementation of the functionality  $\pi_0^\circ$  regardless of the message scheduling. Indeed, fix a computational context  $\varrho\langle\cdot\rangle$  and consider the corresponding chain of compound protocols

$$\varrho\langle\pi^\circ\rangle \xrightarrow{\pi_0^\circ} \varrho\langle\varrho_0\langle\pi_1^\circ\rangle\rangle \equiv \varrho_1\langle\pi_1^\circ\rangle \xrightarrow{\pi_1} \varrho_1\langle\pi_1\rangle , \quad (7.17)$$

where  $\varrho_1\langle\cdot\rangle = \varrho\langle\varrho_0\langle\cdot\rangle\rangle$  is a shorthand for another computational context. Note that we can still complete the corresponding reversed game chain

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \equiv \mathcal{G}_3^{\mathcal{A}_3} \xrightarrow{\text{UC-SEC}} \mathcal{G}_2^{\mathcal{A}_2} \equiv \mathcal{G}_1^{\mathcal{A}_1} \xrightarrow{\text{UC-SEC}} \mathcal{G}_0^{\mathcal{A}_0} \equiv \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} , \quad (7.18)$$

where the security games  $\mathcal{G}_0, \mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3$  correspond to the protocols and contexts from left to right in the chain (7.17) and  $\mathcal{A}_1 = \mathcal{A}_2$  and  $\mathcal{G}_1 = \mathcal{G}_2$ . To be precise, the game chain exists only if the adversarial structures for both protocols  $\pi_0$  and  $\pi_1$  are compatible. Evidently, the result holds also for longer chains of hybrid protocols, provided that all necessary quantitative bounds are compatible.

**Criterion for universal composability.** Although Canetti was the first to define the universal composability for the asynchronous model of computations, the concept itself was known already in the early 1990s. Micali and Rogaway defined and studied the corresponding security objective as *reducibility* in the synchronised model of computations, see articles [MR91a, MR91b] for further discussion. Also, the description of universally composable protocols has been

<sup>3</sup>The latter is another example that illustrates how dangerous it is to define asymptotic security models directly without first considering the exact formalisation.

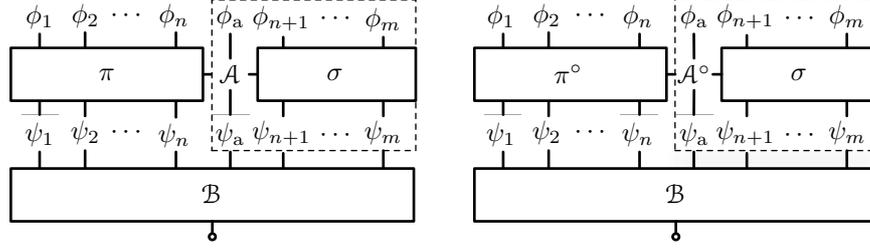


Figure 7.6: Correspondence between parallel execution and stand-alone model. The dashed box represents the compound adversary in the stand-alone model.

implicitly known for decades, starting from the early works [MR91b, Bea91b]. All subsequent works, such as [DM00, PW00, Can01, Lin03b], have just refined technical details. Nevertheless, we provide the characterisation together with a proof sketch, since the latter underlines some intricate connections with sequential composability and emphasises some unexpected details.

Again, we restrict ourselves to the computational models that support errorless decomposition. Consequently, we can ignore the pre- and post-processing phase and concentrate on the interactions between the protocol  $\pi$  and the side-computations  $\sigma$ , see Fig. 7.6. Observe that a real world adversary  $\mathcal{A}$  can always break the conceptual separation between  $\pi$  and  $\sigma$  by selectively transferring data from one process to another. Such behaviour is impossible in the ideal world, where the idealised protocol  $\pi^\circ$  itself is shielded from the adversary  $\mathcal{A}^\circ$ . Note that the proof of Theorem 1 uses just some obvious although very crude bounds on the information flow between these processes. Let  $\mathcal{I}_\sigma$  denote the set of nodes that participate in the side computations. Then the information leakage from the process  $\sigma$  is bounded by the corresponding inputs  $(\phi_i)_{i \in \mathcal{I}_\sigma}$ . Similarly, we can overestimate the reverse flow of information from  $\pi$  to  $\sigma$  by giving the control over the output states  $(\psi_i)_{i \in \mathcal{I}_\sigma}$  to the adversary  $\mathcal{A}^\circ$ .

Of course, the resulting security estimate is very crude, as real world adversaries normally have only partial control over the states  $(\phi_i)_{i \in \mathcal{I}_\sigma}$  and  $(\psi_i)_{i \in \mathcal{I}_\sigma}$ . More precisely, the stand-alone adversary  $\mathcal{A}_s = (\mathcal{A}, \mathcal{P}_i)_{i \in \mathcal{I}_\sigma}$  can be viewed as a tuple, where  $\mathcal{A}$  has no direct control over non-corrupted participants. Hence, we must place structural restrictions to the correspondence  $(\mathcal{B}, \mathcal{A}_s) \mapsto \mathcal{A}_s^\circ$  in the stand-alone model, or otherwise the adversarial structures differ in the real and the ideal world. One possible solution was put forward by Beaver [Bea91b]. Namely, consider a non-rewinding *interface*  $\mathcal{J}$  between the ideal implementation  $\pi^\circ$  and the real world adversary  $\mathcal{A}$  that translates the corresponding communication. Briefly, the interface  $\mathcal{J}$  must simulate all protocol messages for the adversary  $\mathcal{A}$  given only access to the ideal implementation  $\pi^\circ$ . As a result, we can view the interface as a computational context  $\mathcal{J}\langle \cdot \rangle$  that uses the real world adversary  $\mathcal{A}_s$  in a black-box manner to define the corresponding ideal world adversary<sup>4</sup>  $\mathcal{A}_s^\circ = \mathcal{J}\langle \mathcal{A}_s \rangle = \mathcal{J}\langle \mathcal{A}, \mathcal{P}_i \rangle_{i \in \mathcal{I}_\sigma}$ . More formally, we say that a protocol  $\pi$  is  $(t_{\text{re}}, t_{\text{id}}, \varepsilon)$ -close w.r.t. the predicate set  $\mathfrak{B}$  and interface  $\mathcal{J}\langle \cdot \rangle$  in the strong stand-alone model, if for any  $t_{\text{re}}$ -time adversary  $\mathcal{A}_s$  and for any predicate  $\mathcal{B}(\cdot) \in \mathfrak{B}$ , the

<sup>4</sup>Canetti uses a term shell simulator to denote the compound adversary  $\mathcal{J}\langle \mathcal{A}_s \rangle$  in [Can00b].

ideal  $t_{\text{id}}$ -time adversary  $\mathcal{A}_s^\circ = \mathcal{J}\langle\mathcal{A}_s\rangle$  achieves

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}_s} = 1] - \Pr[\mathcal{G}_{\text{real}}^{\mathcal{J}\langle\mathcal{A}_s\rangle} = 1]| \leq \varepsilon \quad (7.19)$$

for any extended input distribution  $\mathfrak{D}_e$ . As the interface  $\mathcal{J}\langle\cdot\rangle$  preserves the adversarial structure, it is straightforward to relax requirements in Theorem 1. However, the proof holds only if the communication of the protocol  $\pi$  is separable from the remaining messages. Here, we assume that all messages of  $\pi$  are routed via a dedicated courier node  $\mathcal{P}_{\text{nw}}^\pi$  but there are many alternatives. For example, the standard formalisation given by Canetti requires that each message contains a special identifier that uniquely determines its purpose [Can00b, p. 21–32].

**Theorem 2.** *A context  $\varrho\langle\cdot\rangle$  and a protocol  $\pi$  that satisfy restrictions (S1)–(S2) preserve strong stand-alone security w.r.t. the interface  $\mathcal{J}\langle\cdot\rangle$ . The corresponding overhead is linear for running times and the advantage does not change.*

*Proof sketch.* For clarity, we first consider a context  $\varrho\langle\cdot\rangle$ , where participants of  $\pi$  do no side-computations. As the restrictions (S1)–(S2) still hold, we can follow the proof of Theorem 1 and split the execution into three phases. In particular, constructive mappings  $\mathcal{A} \mapsto (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$  and  $(\mathcal{A}_1^\circ, \mathcal{A}_2^\circ, \mathcal{A}_3^\circ) \mapsto \mathcal{A}^\circ$  such that

$$\mathcal{G}_{\text{real}}^{\mathcal{A}} \equiv (\mathcal{G}_{\text{real}}^*)^{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3} \quad \text{and} \quad (\mathcal{G}_{\text{ideal}}^*)^{\mathcal{A}_1^\circ, \mathcal{A}_2^\circ, \mathcal{A}_3^\circ} \equiv \mathcal{G}_{\text{ideal}}^{\mathcal{A}^\circ} \quad (7.20)$$

are still the same as in the proof of Theorem 1. Thus, the mapping  $\mathcal{A}_2 \mapsto \mathcal{J}\langle\mathcal{A}_2\rangle$  defines the correspondence used in the proof of Theorem 1 and we obtain

$$|\Pr[(\mathcal{G}_{\text{real}}^*)^{\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3} = 1] - \Pr[(\mathcal{G}_{\text{ideal}}^*)^{\mathcal{A}_1, \mathcal{J}\langle\mathcal{A}_2\rangle, \mathcal{A}_3} = 1]| \leq \varepsilon . \quad (7.21)$$

Recall that  $\mathcal{A}_2$  is actually a compound adversary that captures the behaviour of  $\mathcal{A}$  and non-participants  $(\mathcal{P}_i)_{i \in \mathcal{I}_\sigma}$  during the execution of the protocol  $\pi$ . Since  $\mathcal{J}\langle\mathcal{A}_2\rangle$  does not change the interaction patterns between  $\mathcal{A}$  and non-participants  $(\mathcal{P}_i)_{i \in \mathcal{I}_\sigma}$ , the resulting correspondence  $\mathcal{A} \mapsto \mathcal{A}^\circ$  also preserves the adversarial structure and the claim follows from the proof of Theorem 1.

The general case is slightly more complex, as nodes can concurrently participate in  $\pi$  and  $\sigma$ . Since all protocol messages are routed via the node  $\mathcal{P}_{\text{nw}}^\pi$ , the correspondence  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3) \mapsto (\mathcal{A}_1, \mathcal{J}\langle\mathcal{A}_2\rangle, \mathcal{A}_3)$  described above is formally valid and the bound (7.21) still holds. For the formal proof, we split each node into two virtual sub-nodes  $(\mathcal{P}_i, \mathcal{P}_i^*)$  with a shared<sup>5</sup> internal state, where  $\mathcal{P}_i$  does all the computations except for  $\pi$  and  $\mathcal{P}_i^*$  is reserved for the protocol  $\pi$ . That is, all messages transferred by the dedicated courier node  $\mathcal{P}_{\text{nw}}^\pi$  are sent out and received by sub-nodes  $\mathcal{P}_i^*$ . All other messages are sent and received by normal nodes  $\mathcal{P}_i$ . Also, the adversary  $\mathcal{A}$  can corrupt the node pair  $(\mathcal{P}_i, \mathcal{P}_i^*)$  only simultaneously, as labels  $\mathcal{P}_i$  and  $\mathcal{P}_i^*$  refer to the same node.

As a result, we have reached the simplified case, where participants of  $\pi$  do not engage in any side-computations  $\sigma$ . The claim follows, as the corresponding adversary  $\mathcal{J}\langle\mathcal{A}, \mathcal{P}_1, \dots, \mathcal{P}_n, \mathcal{P}_1^*, \dots, \mathcal{P}_n^*\rangle$  behaves exactly as  $\mathcal{J}\langle\mathcal{A}_2\rangle$ .  $\square$

<sup>5</sup>It is just a mind experiment that gives two labels for the same node to simplify the reasoning in the proof.

**Further relaxations.** First, note that there is no difference between the normal non-corruptible participant  $\mathcal{P}_i$  and a trusted third party  $\mathcal{T}_j$ . Hence, the proof also holds if there are many trusted third parties in the context.

Secondly, we emphasise that the restriction (S2) is essential or otherwise the decomposition can be ill-defined and the reasoning may fail. For example, the lottery protocol  $\pi_{sl}$  in Fig. 7.4 has a non-rewinding interface  $\mathcal{J}\langle\cdot\rangle$  in the strong stand-alone model although it does not preserve the security even in the simple post-processing context described in Section 7.3. Nevertheless, we can use Theorem 2 to construct universally composable protocols that remain secure in all contexts, where the restriction (S1) is satisfied. More precisely, let  $\varrho_0\langle\cdot\rangle$  be a wrapper context that eliminates the possibility of synchronisation errors. For example, a participant in the context  $\varrho_0\langle\cdot\rangle$  starts with  $\pi$  when all participants of  $\pi$  have sent a message “I am ready to start  $\pi$ ” and waits until all participants of  $\pi$  send “I have finished  $\pi$ ” after the completion. Then a modified protocol  $\hat{\pi} = \varrho_0\langle\pi\rangle$  is universally composable, as  $\varrho\langle\hat{\pi}\rangle = \varrho\langle\varrho_0\langle\pi\rangle\rangle = \varrho_1\langle\pi\rangle$  and thus the protocol  $\pi$  in the context  $\varrho_1\langle\cdot\rangle$  satisfies the restrictions (S1)–(S2) by the construction. To be precise, the adversary learns when a participant starts and ends the protocol, but this is usually irrelevant in practice.

Although external synchronisation is always possible, the resulting network delays can cause significant performance penalties. Hence, it makes sense to consider protocols with built-in synchronisation. Such protocols must remain secure even if some protocol messages are received before the protocol execution and assure that final outputs are not released back to the context too early. Note that the proof of Theorem 2 generalises and we can show that a protocol is universally composable if there exists a non-rewinding interface  $\mathcal{J}\langle\cdot\rangle$  such that

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[\mathcal{G}_{\text{real}}^{\mathcal{J}\langle\mathcal{A}\rangle} = 1]| \leq \varepsilon \quad (7.22)$$

for any context  $\varrho\langle\cdot\rangle$  without side-computations. In other words, we must first prove that there exists an interface  $\mathcal{J}\langle\cdot\rangle$  for the stand-alone model and then show that the interface also works for other contexts or alternatively provide some extension to  $\mathcal{J}\langle\cdot\rangle$  that handles the corresponding synchronisation errors.

**Optimality of assumptions.** Obviously, we cannot relax the restriction (S1) without losing all security guarantees. Similarly, a protocol  $\pi$  is universally composable only if there exists a non-rewinding interface  $\mathcal{J}\langle\cdot\rangle$  such that the protocol is secure in the strong stand-alone model w.r.t. this interface  $\mathcal{J}\langle\cdot\rangle$ . For proof, consider a strong stand-alone setting, where the participants  $\mathcal{P}_1, \dots, \mathcal{P}_n$  execute the protocol  $\pi$  and a non-corruptible bystander  $\mathcal{P}_{n+1}$  that interprets the input  $\phi_{n+1}$  as a code and assists the universal adversary  $\mathcal{A}_u$ . More precisely, the adversary  $\mathcal{A}_u$  acts as a wire between the challenger and  $\mathcal{P}_{n+1}$ , i.e.,  $\mathcal{A}_u$  forwards all messages sent by the challenger to  $\mathcal{P}_{n+1}$  and all commands sent by  $\mathcal{P}_{n+1}$  to the challenger. Since the strong stand-alone setting itself is a computational context, there must exist a universal ideal world adversary  $\mathcal{A}_u^\circ$ . Note that  $\mathcal{A}_u^\circ$  is a non-rewinding interface  $\mathcal{J}\langle\cdot\rangle$  that translates communication between  $\mathcal{P}_{n+1}$  and the ideal world challenger. Since  $\mathcal{P}_{n+1}$  can interpret the code of any stand-alone adversary  $\mathcal{A}$  with a linear slowdown, the requirement is optimal up to a constant factor overhead in running times. To be precise, the claim holds only for the synchronised and asynchronous model of computations, since the participant  $\mathcal{P}_{n+1}$  cannot stop the clock for his computations.

## 7.7 TRUSTED SETUP AND UNIVERSAL COMPOSABILITY

The design of universally composable protocols is a challenging task, since the corresponding protocols must have a black-box non-rewinding interface  $\mathcal{J}\langle\cdot\rangle$ . Such requirements diminish possible differences between the adversary and the interface. In fact, the interface  $\mathcal{J}\langle\cdot\rangle$  can be viewed as an attack strategy that allows input extraction and gives partial control over the outputs. Hence, the asymmetry of powers between the adversary and the interface exists only if the adversarial structure itself is asymmetric or there is a significant discrepancy in the running times. In particular, note that classical multi-party protocols with honest majority indeed have non-rewinding interfaces, see for example [Bea91b]. Hence, it is straightforward to show that these protocols are indeed universally composable [Can01, Can00b]. On the other hand, no non-trivial protocol with honest minority can be universally composable unless the interface has a significantly higher computational demands than the adversary [CKL03].

There are two alternatives how to achieve universal composability in the case of honest minority. First, we can allow a significant discrepancy  $t_{\text{re}} \ll t_{\text{id}}$  between the running times of the real and the ideal world adversaries. However, such a choice has several important drawbacks. As the correspondence between real and ideal time-success profiles is less tight, the protocol might provide a way to speed up some computations. Consequently, such protocols are guaranteed to preserve only information-theoretical properties of the corresponding ideal implementations. The large discrepancy  $t_{\text{re}} \ll t_{\text{id}}$  also introduces complications in the security proofs. As a substitution  $\pi \mapsto \pi^\circ$  in a security proof yields a  $t_{\text{id}}$ -time adversary  $\mathcal{A}^\circ$ , a protocol that uses  $\pi$  as a sub-protocol must be secure against  $t_{\text{id}}$ -time adversaries. In particular, we cannot use repetitive substitutions to analyse a context with many instances of the protocol  $\pi$ .

The latter is a serious drawback, since useful sub-protocols are commonly used in many places. Therefore, one needs a stronger version of universal composability that allows to replace many instances of a protocol by a single reduction step, i.e., we must analyse concurrent executions directly. For example, many authors first prove the security of a protocol in the world, where a trusted third party carries out zero-knowledge proofs, and later replace all ideal implementations with a protocol that is known to be secure in such concurrent settings [PR03, Pas04, BS05]. Another more modular alternative was proposed in [PS04] where authors assumed that the interface  $\mathcal{J}\langle\cdot\rangle$  is generally efficient except for few well-described functions  $\mathcal{O}_1, \dots, \mathcal{O}_s$ . Hence, a running time can be viewed as a vector  $\mathbf{t} = (t_0, t_1, \dots, t_s)$ , where  $t_0$  is the number of elementary steps without oracle calls and  $t_i$  is the number of calls to the oracle  $\mathcal{O}_i$ . As the corresponding time bounds  $\mathbf{t}_{\text{re}}$  and  $\mathbf{t}_{\text{id}}$  are more structured than before, the corresponding reduction scheme might withstand many iterations.

A trusted setup is another and often a more practical way to achieve universal composability in the case of honest minority. Such a setup phase naturally introduces necessary asymmetry into the adversarial structure. The trusted dealer  $\mathcal{P}_{\text{ts}}$  is non-corruptible in the real world, whereas the interface  $\mathcal{J}\langle\cdot\rangle$  always corrupts  $\mathcal{P}_{\text{ts}}$  in the ideal world. To be precise, the dealer  $\mathcal{P}_{\text{ts}}$  is not present in the ideal world at all and thus the interface  $\mathcal{J}\langle\cdot\rangle$  must simulate its presence, i.e., the interface  $\mathcal{J}\langle\cdot\rangle$  has full control over  $\mathcal{P}_{\text{ts}}$  by the construction. Such an asymmetry makes the design of universally composable protocols pretty straightforward. Moreover, the

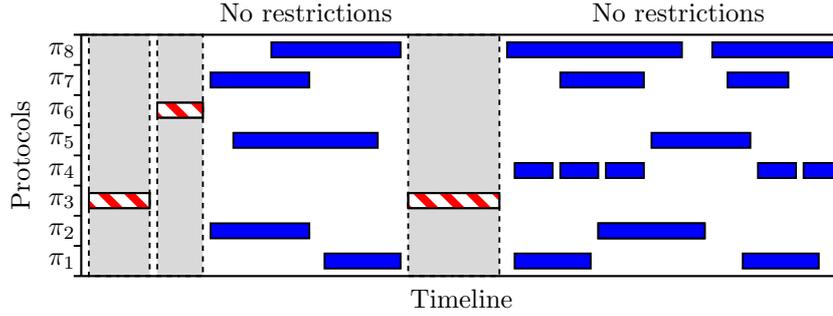


Figure 7.7: Scheduling restrictions posed by modular design. Sub-protocols  $\pi_3$  and  $\pi_6$  are run in isolation, since they implement trusted setup procedures.

corresponding computational overhead compared to stand-alone setting is minimal compared to the protocols with semi-inefficient simulators. Thus, we can systematically use protocols with trusted setup to achieve modularity in security proofs without losing efficiency and round complexity.

The corresponding methodology consists of three steps. First, we must design universally composable protocols for all necessary sub-tasks. At this stage our main design goal is to achieve universal composability with minimal overhead. The corresponding trusted setup phase can be as complicated as needed and the question whether such setup scenario is plausible in practice is irrelevant. In the second step, we use these sub-protocols to construct a round optimal solution. The corresponding security proof is straightforward, since all protocols are universally composable. As a final design step, we replace all trusted setup procedures with standard protocols that are secure only in the strong stand-alone security model. The resulting protocol is both efficient and secure in the strong stand-alone model and has a modular security proof.

To be precise, security in the strong stand-alone model is an understatement, as we can apply Theorem 1 in any computational context. Consequently, the compound protocol remains secure in all computational contexts, where the restrictions (S1)–(S3) and (A1)–(A2) are satisfied during the execution of sub-protocols that implement trusted setup phases, see Fig. 7.7.

However, note that a naive application of these design principles can lead to sub-optimal solutions. As all non-trivial protocols require trusted setup, the resulting protocol can contain an enormous pre-processing phase, i.e., we must still run a large portion of the protocol in isolation. Hence, it is advantageous to reduce the number of setup phases either with bootstrapping or sharing a single setup phase between many protocols. The idea behind bootstrapping is obvious. Imagine that there exists a universally composable protocol  $\pi_{ts}^*$  with a setup phase  $\pi_{ts}^\circ$  that implements more than one setup phase. Then we can iteratively run  $\pi_{ts}^*$  to create enough implementations of the setup phase  $\pi_{ts}^\circ$  as needed. In particular, Canetti and Rabin have shown that common reference string model can be bootstrapped [CR03]. As any functionality can be implemented in the common reference string model [CLOS02], we can always construct a protocol with a short pre-processing stage that creates an initial common reference string. On the other hand, these results show only the feasibility in the polynomial model and may be completely sub-optimal in practice.

Moreover, note that bootstrapping is inherently less efficient than sharing a

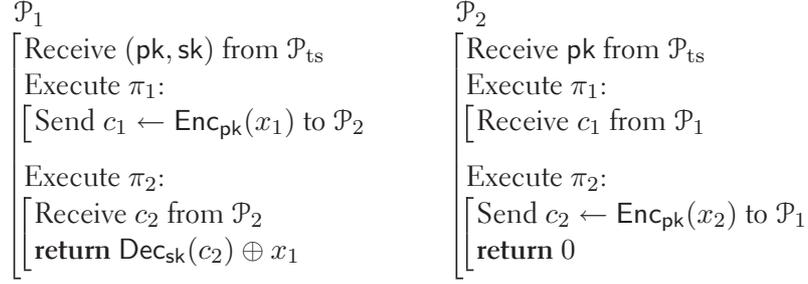
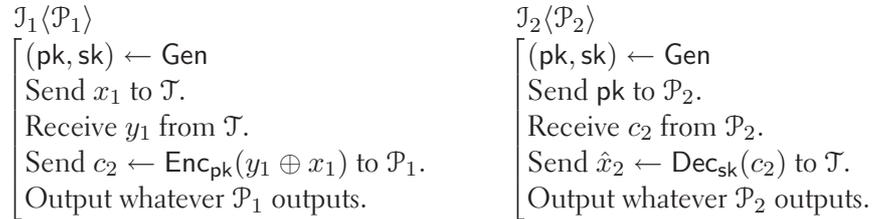


Figure 7.8: Protocols  $\pi_1$  and  $\pi_2$  share a single setup phase and thus fail.

single setup phase  $\pi_{\text{ts}}^\circ$  between different protocols  $\pi_1, \dots, \pi_s$ . However, the gain in efficiency does not come without additional hassle. Namely, we must show that a corresponding compound protocol  $\varrho\langle\pi_1, \dots, \pi_s\rangle$  remains universally composable even if they share the same setup phase  $\pi_{\text{ts}}^\circ$ . Finding such proofs can be a quite complicated and error-prone task, since the messages from different protocols are now correlated and the latter increases the amount of possible attack patterns. As a trivial example, note that one-time pad is a universally composable implementation of encryption functionality as long as the encryption key is chosen uniformly. On the other hand, two-time pad (as a double execution of one-time pad with the same setup phase) is known to be insecure. In other words, universally composable protocols may become insecure if we share the same setup phase between many protocols.

Before going into the technical details, let us first clarify how one should interpret security guarantees in the presence of trusted setup. The latter is not so obvious as it seems. We use a sub-protocol  $\pi_2$  in Fig. 7.8 as an illustrative example. First, note that the protocol  $\pi_2$  uses asymmetric encryption and the trusted dealer  $\mathcal{P}_{\text{ts}}$  establishes an appropriate key pair  $(\mathbf{pk}, \mathbf{sk}) \leftarrow \text{Gen}$ . Second, fix an ideal implementation  $\pi_2^\circ$ , where a trusted third party  $\mathcal{T}$  uses the inputs  $x_1$  and  $x_2$  to compute the outputs  $y_1 = x_1 \oplus x_2$  and  $y_2 = 0$ . Then it is straightforward to show that  $\pi_2$  is universally composable against static adversaries. The corresponding interfaces for malicious participants  $\mathcal{P}_1$  and  $\mathcal{P}_2$  given below



provide a perfect correspondence between the real and ideal world outputs. Note that we can easily find an adversarial strategy  $\mathcal{A}$  such that  $\mathcal{P}_2$  does not know the submitted input  $\hat{x}_2 \leftarrow \text{Dec}_{\mathbf{sk}}(c_2)$ . Still, we can construct the corresponding generic attack, where  $\mathcal{P}_2$  first runs  $\mathcal{J}_2\langle\mathcal{A}\rangle$  in order to extract  $\hat{x}_2$  and then sends  $\hat{c}_2 \leftarrow \text{Enc}_{\mathbf{pk}}(\hat{x}_2)$  to  $\mathcal{P}_1$  and continues the execution of  $\mathcal{J}_2\langle\mathcal{A}\rangle$ . If we consider the averaged performance over all runs of the setup procedure, then both attacks induce the same output distribution. However, if we consider a fixed public key  $\mathbf{pk}$ , then the distributions can have significant differences.

More formally, let  $\omega_{\text{ts}}$  denote the randomness of  $\mathcal{P}_{\text{ts}}$  and let  $\text{Adv}(\mathcal{A}|\omega_{\text{ts}})$  be the corresponding success profile. Then the correspondence  $\mathcal{A} \mapsto \mathcal{J}\langle\mathcal{A}\rangle$  pro-

vides estimates only on the average-case success  $\text{Adv}(\mathcal{A})$ . As the success profiles may have large variance, the average-case bounds are insufficient for estimating  $\text{Adv}(\mathcal{A}|\omega_{\text{ts}})$ . Note that security guarantees w.r.t. a fixed setup run are often more natural in practice. For example, most applications use a single instance of a hash function, such as SHA-1 or WHIRLPOOL, and the security of a public-key infrastructure depends on a single fixed master key. Hence, it makes sense to also consider the universal composability w.r.t. a fixed setup run. Of course, such a setting is meaningful only in the framework of *subjective security* discussed in Section 6.5. In this setting, the interface  $\mathcal{J}(\cdot)$  cannot maliciously corrupt  $\mathcal{P}_{\text{ts}}$ , or otherwise the correspondence  $\mathcal{A} \mapsto \mathcal{J}(\mathcal{A})$  does not preserve closeness between  $\text{Adv}(\mathcal{A}|\omega_{\text{ts}})$  and  $\text{Adv}(\mathcal{A}^\circ|\omega_{\text{ts}})$ . Consequently, the interface  $\mathcal{J}(\cdot)$  can use only the same set of parameters that  $\mathcal{A}$  obtains from  $\mathcal{P}_{\text{ts}}$ , i.e., the trusted setup assures only the correctness and availability of setup parameters.

In the following, we consider only the classical objective setting although the same holds also for the subjective setting. First, it is straightforward to verify that the protocol  $\pi_1$  in Fig. 7.8 is also a universally composable implementation of “do nothing” functionality  $\pi_1^\circ \equiv \perp$ . However, the compound protocol of  $\pi_1$  and  $\pi_2$  with shared setup in Fig. 7.8 is insecure, since a malicious  $\mathcal{P}_2$  can send  $c_1$  back to  $\mathcal{P}_2$  and thus always force the output  $y_1 = x_1 \oplus x_1 = 0$ . Nevertheless, it is easy to prove that concurrent composition  $\varrho\langle\pi_1, \pi_1\rangle$  with the shared setup  $\pi_{\text{ts}}^\circ$  is universally composable and the same is also true for  $\varrho\langle\pi_2, \pi_2\rangle$ . Hence, no simulation strategy  $\mathcal{J}(\cdot)$  for itself makes sharing the setup  $\pi_{\text{ts}}^\circ$  impossible and the failure of  $\varrho\langle\pi_1, \pi_2\rangle$  must be caused by more subtle reasons.

More formally, assume that the protocols  $\pi_1, \dots, \pi_v$  share the same trusted setup phase  $\pi_{\text{ts}}^\circ$ . Then any computational context implicitly defines an inner context  $\varrho_{\text{ts}}\langle\pi_1, \dots, \pi_v\rangle$  that executes only  $\pi_1, \dots, \pi_v$  with the inputs provided by the outer context  $\varrho(\cdot)$ . Since the inner context  $\varrho_{\text{ts}}\langle\pi_1, \dots, \pi_v\rangle$  itself can be viewed as a compound protocol,<sup>6</sup> we must just prove that  $\varrho_{\text{ts}}\langle\pi_1, \dots, \pi_v\rangle$  is universally composable. In particular, note that the final interface  $\mathcal{J}(\cdot)$  for the compound protocol  $\varrho_{\text{ts}}\langle\pi_1, \dots, \pi_v\rangle$  may use a modified setup procedure  $\pi_{\text{ts}}$  in the ideal world and thus the corresponding security proof has three phases. First, we must prove that substituting  $\pi_{\text{ts}}^\circ$  with  $\pi_{\text{ts}}$  does not change the outcome in the real world. In other words, for any valid outer context  $\varrho(\cdot)$

$$|\Pr[\mathcal{G}_{\text{real}}^{\mathcal{A}} = 1] - \Pr[(\mathcal{G}_{\text{real}}^*)^{\mathcal{A}} = 1]| \leq \varepsilon_1, \quad (7.23)$$

where  $\mathcal{G}_{\text{real}}$  and  $\mathcal{G}_{\text{real}}^*$  are the corresponding security games. Secondly, note that direct access to the process  $\pi_{\text{ts}}$  may reveal secret information and thus it may be easier to carry out some computational operations  $\mathcal{O}_1, \dots, \mathcal{O}_s$ . Hence, we should again view time bounds as vectors  $\mathbf{t} = (t_0, \dots, t_s)$ . In the simplest case, all parameters sent out by  $\pi_{\text{ts}}^\circ$  and  $\pi_{\text{ts}}$  are public and no honest participant can carry out these expensive operations  $\mathcal{O}_1, \dots, \mathcal{O}_s$ . In more complicated cases, the setup procedure sends out some private information and the honest participants can also carry out some of these operations  $\mathcal{O}_1, \dots, \mathcal{O}_s$ . As a result, a protocol  $\pi_i$  may become insecure in contexts  $\varrho_{\text{ts}}(\cdot)$ , where such private information is transferred from one node to another. Moreover, the adversary may use honest parties with extra abilities to carry out such operations.

<sup>6</sup>So far we have always assumed that all arguments must be specified in the beginning of the protocol but nothing changes if we allow a gradual demand-based release of arguments.

We can safely and systematically model these aspects by rewriting all protocols  $\pi_1, \dots, \pi_v$  in terms of oracle calls. As a result, we obtain an equivalent computational model, where the protocol  $\pi_{ts}$  broadcasts only public parameters and all running times are expressed as vectors. Since the corresponding reduction schemata quantify time complexity more precisely, it is possible to replace protocols in the inner context  $\varrho_{ts}\langle\pi_1, \dots, \pi_v\rangle$  one by one. Finally, when there are no protocols left, we can complete the proof by removing  $\pi_{ts}$ .

Let us return to the protocol depicted in Fig. 7.8. In this example, we have to count decryption operations as well. Observe that the protocol  $\pi_1$  is universally composable only if the context and the real world adversary do not use decryption operations. As the context contains a single decryption operation, we cannot replace  $\pi_1$  with the ideal implementation  $\pi_1^\circ$ . Although we can replace  $\pi_2$  with the ideal implementation  $\pi_2^\circ$ , the resulting adversary does an extra decryption operation and thus we still cannot replace  $\pi_1$  with  $\pi_1^\circ$ .

Finally, we note that the corresponding security proofs are quite streamlined in all settings, where trusted setup is used to create only public parameters  $\mathit{psp}$ . These security proofs do not use vectors as time bounds, but we still have to take into account the fact that the context can choose the inputs  $\mathbf{x}$  of the protocol based on  $\mathit{psp}$ . Another particularly useful shared setup model is the public-key infrastructure model. However, the latter generally requires precise vectorised time bounds for the reasons shown above. In fact, the presented methodology was largely motivated by the author's own research [GLLM04, LLM05, LLM06, LL07]. These articles study a certain restricted class of protocols that utilise homomorphic encryption. In the final article [LL07], we used a slightly simplified methodology to prove that all these protocols can be constructed by carefully combining three elementary protocols that are universally composable even if they share the corresponding setup phase. Such results are important, as they can significantly simplify practical protocol design and can show what kind of open problems may have significant practical consequences.

## 7.8 PUBLIC INPUTS AND UNIVERSAL COMPOSABILITY

Recall that no protocol without trusted setup can remain universally composable and preserve privacy of inputs in the presence of honest minority [CKL03]. Superficially, it seems to imply that no useful protocol with honest minority can be universally composable in the standard model. However, such an impression is misleading, since some important practical functionalities indeed reveal all inputs. For example, common primitives for authentication, such as message authentication, tamper-resist storage and public broadcast, do reveal all inputs. Consequently, a study of universally composable protocols with public inputs is not a mere theoretical exercise, but also has practical consequences.

More formally, we say that the inputs of a protocol  $\pi$  are *public* if all participants first submit their inputs to the ideal world adversary and then continue interaction with the trusted third party as usual. A closer inspection reveals that all universally composable protocols with honest minority must have public inputs and thus the definition is optimal, see the results in [CKL03]. Moreover, protocols with public inputs cannot generally preserve universal composability and implement a non-deterministic functionality, see again [CKL03] for precise

reasoning. For less technical argumentation, recall that any functionality can be implemented in the trusted setup model [CLOS02]. Hence, a trusted setup procedure itself cannot have universally composable implementation in the presence of honest minority, otherwise every function would have a universally composable implementation. The claim follows, as trusted setup procedures have no inputs and are thus protocols with public inputs.

Obviously, a protocol  $\pi$  with public inputs reveals all outputs when the ideal functionality is deterministic. As a result, we can guarantee only the correctness of outputs. Secondly, note that an interface  $\mathcal{J}_{\text{sim}}\langle\cdot\rangle$  can straightforwardly simulate the execution of the protocol  $\pi$  to the real world adversary  $\mathcal{A}$ , as all inputs of honest nodes are available to  $\mathcal{J}_{\text{sim}}\langle\cdot\rangle$  in the ideal world.<sup>7</sup> Now the correspondence between real and ideal world is complete as soon as we can extract the inputs of corrupted nodes from the traffic between  $\mathcal{A}$  and  $\mathcal{J}_{\text{sim}}\langle\cdot\rangle$ .

More formally, let  $\text{Extr}(\cdot)$  be a  $t_{\text{ex}}$ -time function that takes in the traffic between  $\mathcal{A}$  and  $\mathcal{J}_{\text{sim}}\langle\cdot\rangle$  and let  $\hat{\mathbf{x}} = (\hat{x}_1, \dots, \hat{x}_n)$  be the corresponding guessed input. Then the input extraction fails for a single round functionality if  $x_i \neq \hat{x}_i$  or  $y_i \neq f_i(\hat{\mathbf{x}})$  for uncorrupted participants  $\mathcal{P}_i$ . For a multi-round functionality,  $\text{Extr}(\cdot)$  must produce guesses for each round and  $\text{Extr}(\cdot)$  fails if it fails for any round. A protocol  $\pi$  is  $(t_{\text{re}}, t_{\text{ex}}, t_{\text{cnt}}, \varepsilon)$ -extractable if the input extraction fails with the probability at most  $\varepsilon$ . The following theorem describes when a protocol with honest minority is universally composable.

**Theorem 3.** *There exists a constant  $c > 0$  such that a  $(t_{\text{re}}, t_{\text{ex}}, t_{\text{cnt}}, \varepsilon)$ -extractable protocol  $\pi$  with public inputs is  $(t_{\text{re}}, c \cdot t_{\text{re}}, t_{\text{cnt}}, \varepsilon)$ -universally composable.*

*Proof sketch.* Clearly, the simulating interface  $\mathcal{J}_{\text{sim}}\langle\cdot\rangle$  together with extraction function  $\text{Extr}(\cdot)$  is the necessary interface  $\mathcal{J}\langle\cdot\rangle$ . The claim follows, as the outputs of  $\pi$  and  $\pi^\circ$  differ only if the extraction fails.  $\square$

**Further comments.** There is a strange duality between Theorem 1 and 3. Namely, the proofs of these theorems handle the information flow between a protocol  $\pi$  and side-computations  $\sigma$  in a similar fashion. In the proof of Theorem 1, we corrupt all non-participants of the protocol  $\pi$  and thus manage to reduce the security to the stand-alone setting. More precisely, we first learn the inputs of non-participants to incorporate the process  $\sigma$  into the ideal world adversary and finally replace the outputs of non-participants to assure the correspondence between the real and the ideal world. In the proof of Theorem 3, we do exactly the opposite. That is, we first learn inputs of honest participants to simulate  $\pi$  in the ideal world, and then use the input extraction to assure correspondence between the real and the ideal world. The resulting security guarantees are stronger only because  $\text{Extr}(\cdot)$  is a non-rewinding algorithm.

**Message authentication.** Message authentication protocols form the most prominent class of protocols with public inputs. Recall that the main aim of these protocols is to transfer inputs between the nodes  $\mathcal{P}_1, \dots, \mathcal{P}_n$  by sending messages through a malicious courier node  $\mathcal{P}_{\text{nw}}$ . Consequently, the input extraction itself is trivial, we just have to follow the protocol instructions. Moreover, the input extraction fails only if the adversary succeeds in deception. As a result,

<sup>7</sup>To be precise, this claim does not hold for exotic computational models, where missing real world messages alter the message order in the ideal world.

most of the authentication protocols are automatically universally composable, as soon as they satisfy classical security requirements. As an example, consider unilateral and cross-authentication protocols between honest  $\mathcal{P}_1$  and  $\mathcal{P}_2$ . Let  $x_1$  and  $x_2$  be the corresponding inputs, then both participants should learn  $x_1$  in the unilateral and  $(x_1, x_2)$  in the cross-authentication protocol. Commonly, the security of a protocol is defined through a game in the asynchronous model of computation, where the adversary provides inputs to both parties and succeeds in deception if the outputs differ from ideal implementation. For unilateral authentication protocols, the adversary succeeds if  $\mathcal{P}_2$  reaches accepting state and outputs  $\hat{x}_1 \neq x_1$ . For cross-authentication protocols, the adversary succeeds if either  $\mathcal{P}_1$  or  $\mathcal{P}_2$  reaches accepting state and outputs  $(\hat{x}_1, \hat{x}_2) \neq (x_1, x_2)$ .

Any computational context  $\rho(\cdot)$  together with an adversary  $\mathcal{A}$  gives a rise to a new compound adversary  $\mathcal{A}^*$  that plays the original security game. Moreover, the extraction function indeed fails only if the compound adversary  $\mathcal{A}^*$  succeeds in deception. Hence, classical message authentication protocols are automatically universally composable. The result holds even if we allow the adversary to corrupt participants, but then we have to redefine what a deception failure means. Analogous results hold also for group authentication protocols.

Note that authentication protocols also have a trusted setup phase, where the secret keys are created and transferred to all participants. Thus, a parallel execution of several message authentication protocols that share the same setup phase may be completely insecure. One possible solution is to use bootstrapping. For example, it is straightforward to use pseudorandom functions to generate many computationally independent session keys from a single master key. Secondly, we can study the extractability of a compound protocol  $\rho_{\text{ts}}(\pi_1, \dots, \pi_s)$  with shared setup  $\pi_{\text{ts}}^\circ$  directly. This approach is commonly known as a Bellare-Rogaway model for message authentication, see [BR93a, BR95]. In the corresponding model, deception is again defined so that adversary succeeds in deception if the corresponding input extraction procedure fails. As a result, a concurrent composition of authentication protocols is universally composable as soon as it is secure in the Bellare-Rogaway model.

In recent years, cryptographers have also investigated manual authentication protocols [Vau05, PV06a, PV06b, LN06, LP08]. These protocols do not have a complex trusted setup phase. All of them can be implemented either in the standard or in the common reference string model. The authenticity of messages is achieved by sending short authenticated strings. As a result, one can easily show that a concurrent execution of these protocols is still universally composable even if they share the setup phase. However, the latter is still insufficient for security, since short authenticated strings are sent without tags that identify the corresponding protocol instances. Consequently, we must additionally assure that the adversary cannot switch short authenticated strings between the protocols, otherwise the corresponding synchronisation errors may invalidate the composability guarantees. Thus, we must either use further analysis to determine the maximal achievable damage caused by clever message switching as in [Vau05, PV06a, PV06b], or alternatively add further usage restrictions that eliminate the possibility of such attacks as in [LN06, LP08].

## BIBLIOGRAPHY

- [AD97] Miklós Ajtai and Cynthia Dwork. A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence. In *Proceedings of STOC 1997*, pages 284–293. ACM Press, 1997.
- [AFK<sup>+</sup>07] Kazumaro Aoki, Jens Franke, Thorsten Kleinjung, Arjen Lenstra, and Dag Arne Osvik. A kilobit special number field sieve factorization. Cryptology ePrint Archive, Report 2007/205, 2007. Available from <http://eprint.iacr.org/>.
- [AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In Birgit Pfitzmann, editor, *Proceedings of EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135. Springer, 2001.
- [AL07] Yonatan Aumann and Yehuda Lindell. Security Against Covert Adversaries: Efficient Protocols for Realistic Adversaries. In Salil P. Vadhan, editor, *Proceedings of TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 137–156. Springer, 2007.
- [ASW98] N. Asokan, Victor Shoup, and Michael Waidner. Optimistic Fair Exchange of Digital Signatures (Extended Abstract). In Kaisa Nyberg, editor, *Proceedings of EUROCRYPT 1998*, volume 1403 of *Lecture Notes in Computer Science*, pages 591–606. Springer, 1998.
- [BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem. In Christian Cachin and Jan Camenisch, editors, *Proceeding of EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2004.
- [BC86] Gilles Brassard and Claude Crépeau. Non-Transitive Transfer of Confidence: A Perfect Zero-Knowledge Interactive Protocol for SAT and Beyond. In *Proceedings of FOCS 1986*, pages 188–195. IEEE Computer Society, 1986.
- [BDJR97] Mihir Bellare, Anand Desai, E. Jökipii, and Phillip Rogaway. A Concrete Security Treatment of Symmetric Encryption. In *Proceedings of FOCS 1997*, pages 394–403. IEEE Computer Society, 1997. The extended version of the article is available from <http://www-cse.ucsd.edu/users/mihir/papers/sym-enc.html>.
- [Bea91a] Donald Beaver. Foundations of Secure Interactive Computing. In Joan Feigenbaum, editor, *Proceedings of CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 377–391. Springer, 1991.
- [Bea91b] Donald Beaver. Secure Multiparty Protocols and Zero-Knowledge Proof Systems Tolerating a Faulty Minority. *Journal of Cryptology*, 4(2):75–122, 1991.

- [BG89] Donald Beaver and Shafi Goldwasser. Multiparty Computation with Faulty Majority. In Gilles Brassard, editor, *Proceedings of CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 589–590. Springer, 1989.
- [BG92] Mihir Bellare and Oded Goldreich. On Defining Proofs of Knowledge. In Ernest F. Brickell, editor, *Proceeding of CRYPTO 1992*, volume 740 of *Lecture Notes in Computer Science*, pages 390–420. Springer, 1992.
- [Bil95] Patrick Billingsley. *Probability and Measure*. Probability and Mathematical Statistics. Wiley, third edition edition, 1995.
- [Bin00] N. H. Bingham. Studies in the history of probability and statistics XLVI. Measure into probability: From Lebesgue to Kolmogorov. *Biometrika*, 87(1):145–156, 2000.
- [BK04] Ian F. Blake and Vladimir Kolesnikov. Strong Conditional Oblivious Transfer and Computing on Intervals. In Pil Joong Lee, editor, *Proceedings of ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 515–529. Springer, 2004.
- [BKR94] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The Security of Cipher Block Chaining. In Yvo Desmedt, editor, *Proceedings of CRYPTO 1994*, volume 839 of *Lecture Notes in Computer Science*, pages 341–358. Springer, 1994.
- [BL04] Boaz Barak and Yehuda Lindell. Strict Polynomial-Time in Simulation and Extraction. *SIAM Journal on Computing*, 33(4):738–818, 2004.
- [BL06] Ahto Buldas and Sven Laur. Do Broken Hash Functions Affect the Security of Time-Stamping Schemes? In Jianying Zhou, Moti Yung, and Feng Bao, editors, *Proceedings of ACNS 2006*, volume 3989 of *Lecture Notes in Computer Science*, pages 50–65. Springer, 2006.
- [BL07] Ahto Buldas and Sven Laur. Knowledge-Binding Commitments with Applications in Time-Stamping. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Proceedings of PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 150–165. Springer, 2007.
- [Bla01] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proceedings of CSFW 2001*, pages 82–96. IEEE Computer Society, 2001.
- [Bla06a] John Black. The Ideal-Cipher Model, Revisited: An Uninstantiable Blockcipher-Based Hash Function. In Matthew J. B. Robshaw, editor, *Proceedings of FSE 2006*, volume 4047 of *Lecture Notes in Computer Science*, pages 328–340. Springer, 2006.

- [Bla06b] Bruno Blanchet. A Computationally Sound Mechanized Prover for Security Protocols. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 140–154. IEEE Computer Society, 2006.
- [Bla07] Bruno Blanchet. Computationally Sound Mechanized Proofs of Correspondence Assertions. In *20th IEEE Computer Security Foundations Symposium*, 2007. To appear.
- [BM89] Mihir Bellare and Silvio Micali. Non-Interactive Oblivious Transfer and Applications. In Gilles Brassard, editor, *Proceedings of CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 547–557. Springer, 1989.
- [BMR90] Donald Beaver, Silvio Micali, and Phillip Rogaway. The Round Complexity of Secure Protocols (Extended Abstract). In *Proceedings of STOC 1990*, pages 503–513. ACM Press, 1990.
- [BOCG93] Michael Ben-Or, Ran Canetti, and Oded Goldreich. Asynchronous secure computation. In *Proceedings of STOC 1993*, pages 52–61. ACM Press, 1993.
- [BOGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation (Extended Abstract). In *Proceedings of STOC 1988*, pages 1–10. ACM Press, 1988.
- [Bor62] Emile Borel. *Probabilities and life*. Dover Publications, 1962. Translated from the French by Maurice Baudin.
- [BP06] Bruno Blanchet and David Pointcheval. Automated Security Proofs with Sequences of Games. In Cynthia Dwork, editor, *Proceedings of CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 537–554. Springer, 2006.
- [BPW04] Michael Backes, Birgit Pfitzmann, and Michael Waidner. The Reactive Simulatability (RSIM) Framework for Asynchronous Systems. Cryptology ePrint Archive, Report 2004/082, 2004. Available from <http://eprint.iacr.org/>.
- [BR93a] Mihir Bellare and Phillip Rogaway. Entity Authentication and Key Distribution. In Douglas R. Stinson, editor, *Proceedings of CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 1993.
- [BR93b] Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *Proceedings of ACM CCS 1993*, pages 62–73. ACM Press, 1993.
- [BR95] Mihir Bellare and Phillip Rogaway. Provably secure session key distribution: the three party case. In *Proceedings of STOC 1995*, pages 57–66. ACM Press, 1995.

- [BR04] Mihir Bellare and Phillip Rogaway. Code-Based Game-Playing Proofs and the Security of Triple Encryption. Cryptology ePrint Archive, Report 2004/331, 2004. Available form <http://eprint.iacr.org/>.
- [BR06] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *Proceedings of EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426. Springer, 2006.
- [BS04] Ahto Buldas and Märt Saarepera. On Provably Secure Time-Stamping Schemes. In *Proceedings of ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, pages 500–514. Springer, 2004.
- [BS05] Boaz Barak and Amit Sahai. How To Play Almost Any Mental Game Over The Net - Concurrent Composition via Super-Polynomial Simulation. In *Proceedings of FOCS 2005*, pages 543–552. IEEE Computer Society, 2005.
- [BT92] George Edward Pelham Box and George C. Tiao. *Bayesian Inference in Statistical Analysis*. Wiley Classics Library. Wiley, 1992.
- [Can00a] Ran Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000.
- [Can00b] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Cryptology ePrint Archive, Report 2000/067, 2000. Available form <http://eprint.iacr.org/>.
- [Can01] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Proceedings of FOCS 2001*, pages 136–145. IEEE Computer Society, 2001.
- [CB02] George Casella and Roger L. Berger. *Statistical Inference*. Advanced series. Duxbury Press, second edition edition, 2002.
- [CC00] Christian Cachin and Jan Camenisch. Optimistic Fair Secure Computation. In Mihir Bellare, editor, *Proceedings of CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 93–111. Springer, 2000.
- [CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty Unconditionally Secure Protocols (Extended Abstract). In *Proceedings of STOC 1988*, pages 11–19. ACM Press, 1988.
- [CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively Secure Multi-Party Computation. In *Proceedings of STOC 1996*, pages 639–648. ACM Press, 1996.

- [CGH04a] Ran Canetti, Oded Goldreich, and Shai Halevi. On the Random-Oracle Methodology as Applied to Length-Restricted Signature Schemes. In Moni Naor, editor, *Proceedings of TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 40–57. Springer, 2004.
- [CGH04b] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited. *Journal of the ACM*, 51(4):557–594, 2004.
- [Cha86] David Chaum. Demonstrating That a Public Predicate Can Be Satisfied Without Revealing Any Information About How. In Andrew M. Odlyzko, editor, *Proceedings of CRYPTO 1986*, volume 263 of *Lecture Notes in Computer Science*, pages 195–199. Springer, 1986.
- [Chu36] Alonzo Church. An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, 58(2):345–363, April 1936.
- [CK89] Benny Chor and Eyal Kushilevitz. A zero-one law for Boolean privacy. In *Proceedings of STOC 1989*, pages 62–72, New York, NY, USA, 1989. ACM Press.
- [CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the Limitations of Universally Composable Two-Party Computation without Set-up Assumptions. In Eli Biham, editor, *Proceedings of EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 68–86. Springer, 2003.
- [Cle86] Richard Cleve. Limits on the Security of Coin Flips when Half the Processors Are Faulty (Extended Abstract). In *Proceedings of STOC 1986*, pages 364–369. ACM Press, 1986.
- [CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proceedings of STOC 2002*, pages 494–503. ACM Press, 2002.
- [CO99] Ran Canetti and Rafail Ostrovsky. Secure Computation with Honest-Looking Parties: What If Nobody Is Truly Honest? (Extended Abstract). In *Proceedings of STOC 1999*, pages 255–264. ACM Press, 1999.
- [Cob64] Alan Cobham. The intrinsic computational difficulty of functions. In Yehoshua Bar-Hillel, editor, *Proceedings of 1964 International Congress for Logic, Methodology, and Philosophy of Sciences*, pages 24–30, Amsterdam, 1964. North-Holland Publishing Company.
- [CR72] Stephen A. Cook and Robert A. Reckhow. Time-bounded random access machines. In *Proceedings of STOC 1972*, pages 73–80, New York, NY, USA, 1972. ACM Press.

- [CR03] Ran Canetti and Tal Rabin. Universal Composition with Joint State. In Dan Boneh, editor, *Proceedings of CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 265–281. Springer, 2003.
- [CV01] Yannick Chevalier and Laurent Vigneron. A Tool for Lazy Verification of Security Protocols. In *Proceedings of ASE 2001*, pages 373–376. IEEE Computer Society, 2001.
- [Den02] Alexander W. Dent. Adapting the Weaknesses of the Random Oracle Model to the Generic Group Model. In Yuliang Zheng, editor, *Proceedings of ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 100–109. Springer, 2002.
- [Den06] Alexander W. Dent. A Note On Game-Hopping Proofs. Cryptology ePrint Archive, Report 2006/260, 2006. Available from <http://eprint.iacr.org/>.
- [DK02] Ivan Damgård and Maciej Koprowski. Generic Lower Bounds for Root Extraction and Signature Schemes in General Groups. In Lars R. Knudsen, editor, *Proceedings of EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 256–271. Springer, 2002.
- [DKMR05] Anupam Datta, Ralf Küsters, John C. Mitchell, and Ajith Ramanathan. On the Relationships Between Notions of Simulation-Based Security. In Joe Kilian, editor, *Proceedings of TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 476–494. Springer, 2005.
- [DM00] Yevgeniy Dodis and Silvio Micali. Parallel Reducibility for Information-Theoretically Secure Computation. In Mihir Bellare, editor, *Proceedings of CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 74–92. Springer, 2000.
- [DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *Journal of the ACM*, 51(6):851–898, 2004.
- [Edm65] Jack Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.
- [FFS87] Uriel Fiege, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. In *Proceedings of STOC 1987*, pages 210–217, New York, NY, USA, 1987. ACM Press.
- [Fie92] Stephen E. Fienberg. A brief history of statistics in three and one-half chapters: A review essay. *Statistical Science*, 7(2):208–225, 1992.
- [Fie05] Stephen A. Fienberg. When did Bayesian Inference become "Bayesian"? *Bayesian Analysis*, 1(1):1–40, July 2005.

- [FIM<sup>+</sup>06] Joan Feigenbaum, Yuval Ishai, Tal Malkin, Kobbi Nissim, Martin J. Strauss, and Rebecca N. Wright. Secure multiparty computation of approximations. *ACM Transactions on Algorithms*, 2(3):435–472, 2006.
- [FIPR05] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword Search and Oblivious Pseudorandom Functions. In Joe Kilian, editor, *Proceedings of TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 303–324. Springer, 2005.
- [Fis01] Marc Fischlin. A Cost-Effective Pay-Per-Multiplication Comparison Method for Millionaires. In David Naccache, editor, *Proceedings of CT-RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 457–472. Springer, 2001.
- [FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In Christian Cachin and Jan Camenisch, editors, *Proceedings of EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2004.
- [For87] Lance Fortnow. The Complexity of Perfect Zero-Knowledge (Extended Abstract). In *Proceedings of STOC 1987*, pages 204–209. ACM Press, 1987.
- [FS89] Uriel Feige and Adi Shamir. Zero Knowledge Proofs of Knowledge in Two Rounds. In Gilles Brassard, editor, *Proceedings of CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 526–544. Springer, 1989.
- [FY92] Matthew K. Franklin and Moti Yung. Communication Complexity of Secure Computation (Extended Abstract). In *Proceedings of STOC 1992*, pages 699–710. ACM Press, 1992.
- [GK96a] Oded Goldreich and Ariel Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.
- [GK96b] Oded Goldreich and Hugo Krawczyk. On the Composition of Zero-Knowledge Proof Systems. *SIAM Journal on Computing*, 25(1):169–192, 1996.
- [GK03] Shafi Goldwasser and Yael Tauman Kalai. On the (In)security of the Fiat-Shamir Paradigm. In *Proceedings of FOCS 2003*, pages 102–113. IEEE Computer Society, 2003.
- [GL90] Shafi Goldwasser and Leonid A. Levin. Fair Computation of General Functions in Presence of Immoral Majority. In Alfred Menezes and Scott A. Vanstone, editors, *Proceedings of CRYPTO 1990*, volume 537 of *Lecture Notes in Computer Science*, pages 77–93. Springer, 1990.

- [GLLM04] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On Private Scalar Product Computation for Privacy-Preserving Data Mining. In Choonsik Park and Seongtaek Chee, editors, *Proceedings of ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, pages 104–120. Springer, 2004.
- [GM82] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing (STOC)*, pages 365–377. ACM Press, 1982.
- [GM84] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
- [GMR85] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of STOC 1985*, pages 291–304. ACM Press, 1985.
- [GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.
- [GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to Play any Mental Game or A Completeness Theorem for Protocols with Honest Majority. In *Proceedings of STOC 1987*, pages 218–229. ACM Press, 1987.
- [GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):690–728, 1991.
- [GO94] Oded Goldreich and Yair Oren. Definitions and Properties of Zero-Knowledge Proof Systems. *Journal of Cryptology*, 7(1):1–32, 1994.
- [Gol98] Oded Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Algorithms and Combinatorics. Springer, 1998.
- [Gol04] Oded Goldreich. *Foundations of Cryptography II: Basic Applications*. Cambridge University Press, 2004.
- [Gol07] Oded Goldreich. On Expected Probabilistic Polynomial-Time Adversaries: A Suggestion for Restricted Definitions and Their Benefits. In Salil P. Vadhan, editor, *Proceedings of TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 174–193. Springer, 2007. See also the corresponding ECCC report.
- [GRR98] Rosario Gennaro, Michael O. Rabin, and Tal Rabin. Simplified VSS and Fact-Track Multiparty Computations with Applications to Threshold Cryptography. In *Proceedings of PODC 1998*, pages 101–111. ACM Press, 1998.

- [Hal05] Shai Halevi. A plausible approach to computer-aided cryptographic proofs. Cryptology ePrint Archive, Report 2005/181, 2005. Available from <http://eprint.iacr.org/>.
- [Hil92] Alain P. Hiltgen. Constructions of Freely-One-Way Families of Permutations. In Jennifer Seberry and Yuliang Zheng, editors, *Proceedings of ASIACRYPT 1992*, volume 718 of *Lecture Notes in Computer Science*, pages 422–434. Springer, 1992.
- [HL92] Amir Herzberg and Michael Luby. Pseudorandomness in Cryptography. In Ernest F. Brickell, editor, *Proceedings of CRYPTO 1992*, volume 740 of *Lecture Notes in Computer Science*, pages 421–432. Springer, 1992.
- [HMU00] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 2000.
- [Imp95] Russell Impagliazzo. A Personal View of Average-Case Complexity. In *Structure in Complexity Theory Conference*, pages 134–147, 1995.
- [IR88] Russell Impagliazzo and Steven Rudich. Limits on the Provable Consequences of One-way Permutations. In Shafi Goldwasser, editor, *Proceedings of CRYPTO 1988*, volume 403 of *Lecture Notes in Computer Science*, pages 8–26. Springer, 1988.
- [Jay03] Edwin T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.
- [Jef04] Richard C. Jeffrey. *Subjective Probability the Real Thing*. Cambridge University Press, 2004.
- [KL05] Jonathan Katz and Yehuda Lindell. Handling Expected Polynomial-Time Strategies in Simulation-Based Security Proofs. In Joe Kilian, editor, *Proceedings of TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 128–149. Springer, 2005.
- [Kle43] Stephen. C. Kleene. Recursive Predicates and Quantifiers. *Transactions of the American Mathematical Society*, 53(1):41–73, January 1943.
- [KLL06] Emilia Käsper, Sven Laur, and Helger Lipmaa. Black-Box Knowledge Extraction Revisited: Universal Approach with Precise Bounds. Cryptology ePrint Archive, Report 2006/356, 2006. Available from <http://eprint.iacr.org/>.
- [KO04] Jonathan Katz and Rafail Ostrovsky. Round-Optimal Secure Two-Party Computation. In Matthew K. Franklin, editor, *Proceedings of CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 335–354. Springer, 2004.

- [KS05] Lea Kissner and Dawn Xiaodong Song. Privacy-Preserving Set Operations. In Victor Shoup, editor, *Proceedings of CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 241–257. Springer, 2005.
- [Lan61] Rolf Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5:183–191, 1961.
- [Lin03a] Yehuda Lindell. Bounded-concurrent secure two-party computation without setup assumptions. In *Proceedings of STOC 2003*, pages 683–692. ACM Press, 2003.
- [Lin03b] Yehuda Lindell. General Composition and Universal Composability in Secure Multi-Party Computation. In *Proceedings of FOCS 2003*, pages 394–403. IEEE Computer Society, 2003.
- [Lin04] Yehuda Lindell. Lower Bounds for Concurrent Self Composition. In Moni Naor, editor, *Proceedings of TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 203–222. Springer, 2004.
- [Lip05] Helger Lipmaa. An Oblivious Transfer Protocol with Log-Squared Communication. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *Proceedings of ISC 2005*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer, 2005.
- [LL06] Sven Laur and Helger Lipmaa. Consistent adaptive two-party computations. Cryptology ePrint Archive, Report 2006/088, 2006. Available from <http://eprint.iacr.org/>.
- [LL07] Sven Laur and Helger Lipmaa. A New Protocol for Conditional Disclosure of Secrets and Its Applications. In Jonathan Katz and Moti Yung, editors, *Proceedings of ACNS 2007*, volume 4521 of *Lecture Notes in Computer Science*, pages 207–225. Springer, 2007.
- [LLM05] Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. Private Itemset Support Counting. In Sihan Qing, Wenbo Mao, Javier Lopez, and Guilin Wang, editors, *Proceedings of ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, pages 97–111. Springer, 2005.
- [LLM06] Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. Cryptographically private support vector machines. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of KDD 2006*, pages 618–624. ACM Press, 2006.
- [LMMS98] Patrick Lincoln, John C. Mitchell, Mark Mitchell, and Andre Scedrov. A probabilistic poly-time framework for protocol analysis. In *Proceedings of ACM CCS 1998*, pages 112–121, New York, NY, USA, 1998. ACM Press.

- [LMR83] Michael Luby, Silvio Micali, and Charles Rackoff. How to Simultaneously Exchange a Secret Bit by Flipping a Symmetrically-Biased Coin. In *Proceedings of FOCS 1983*, pages 11–21. IEEE Computer Society, 1983.
- [LN06] Sven Laur and Kaisa Nyberg. Efficient Mutual Data Authentication Using Manually Authenticated Strings. In David Pointceval, Yi Mu, and Kefei Chen, editors, *Proceedings of CANS 2006*, volume 4301 of *Lecture Notes in Computer Science*, pages 90–107. Springer, 2006.
- [Low97] Gavin Lowe. Casper: A Compiler for the Analysis of Security Protocols. In *Proceedings OF CSFW 1997*, pages 18–30. IEEE Computer Society, 1997.
- [LP04] Yehuda Lindell and Benny Pinkas. A Proof of Yao’s Protocol for Secure Two-Party Computation. Cryptology ePrint Archive, Report 2004/175, 2004. Available from <http://eprint.iacr.org/>.
- [LP07] Yehuda Lindell and Benny Pinkas. An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries. In Moni Naor, editor, *Proceedings of EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 52–78. Springer, 2007.
- [LP08] Sven Laur and Sylvain Pasini. SAS-Based Group Authentication and Key Agreement Protocols. In *Proceedings of PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*, pages 197–213. Springer, 2008.
- [LR06] Gregor Leander and Andy Rupp. On the Equivalence of RSA and Factoring Regarding Generic Ring Algorithms. In Xuejia Lai and Kefei Chen, editors, *Proceedings of ASIACRYPT 2006*, volume 4284 of *Lecture Notes in Computer Science*, pages 241–251. Springer, 2006.
- [Lub96] Michael Luby. *Pseudorandomness and Cryptographic Applications*. Princeton University Press, 1996.
- [LV97] Ming Li and Paul Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*. Graduate Texts in Computer Science. Springer, 2nd edition edition, 1997.
- [Mas96] James L. Massey. The difficulty with difficulty. IACR Distinguished Lecture at Eurocrypt 1996, 1996. Available from IACR webpage <http://www.iacr.org/conferences/ec96/massey.html>.
- [Mau02] Ueli M. Maurer. Indistinguishability of Random Systems. In Lars R. Knudsen, editor, *Proceedings of EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 110–132. Springer, 2002.

- [Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Communications of the ACM*, 21(4):294–299, 1978.
- [Mor98] Hans Moravec. *Robot: Mere Machine to Transcendent Mind*, chapter Chapter 3: Power and Presence. Oxford University Press, 1998. Computational power of various computing devices measured in MIPS. Missing data points for years 2000 and 2005 are filled with the data from TOP500 project. See <http://www.top500.org/>. The conversion from Flops to MIPS was done by taking 1 MIPS = 3.2MFlops.
- [MR91a] Silvio Micali and Phillip Rogaway. Secure Computation (Abstract). In Joan Feigenbaum, editor, *Proceedings of CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 392–404. Springer, 1991.
- [MR91b] Silvio Micali and Phillip Rogaway. Secure Computation (Preliminary Report). Technical Report MIT-LCS-TR-511, Massachusetts Institute of Technology, 1991.
- [MRH04] Ueli M. Maurer, Renato Renner, and Clemens Holenstein. Indifferentiability, Impossibility Results on Reductions, and Applications to the Random Oracle Methodology. In Moni Naor, editor, *Proceedings of TCC 2004*, volume 2951 of *Lecture Notes in Computer Science*, pages 21–39. Springer, 2004.
- [MRS88] Silvio Micali, Charles Rackoff, and Bob Sloan. The Notion of Security for Probabilistic Cryptosystems. *SIAM Journal on Computing*, 17(2):412–426, 1988.
- [MRST06] John C. Mitchell, Ajith Ramanathan, Andre Scedrov, and Vanessa Teague. A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. *Theoretical Computer Science*, 353(1-3):118–164, 2006.
- [Nat00] National Institute of Standards and Technology. Digital Signature Standard. FIPS PUB 186-2, U.S. Department of Commerce, January 2000. Available from <http://csrc.nist.gov/publications/.../fips/fips186-2/fips186-2-change1.pdf>.
- [Nec94] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994. Translated from *Matematicheskie Zametki*, Vol. 55, No. 2, pp. 91–101, February, 1994.
- [Net06] Network Working Group. The Transport Layer Security (TLS) Protocol Version 1.1. RFC 4346, The Internet Engineering Task Force, April 2006. Available from <http://tools.ietf.org/html/rfc4346>.

- [NN92] Hanne Riis Nielson and Flemming Nielson. *Semantics with applications: a formal introduction*. John Wiley & Sons, Inc., New York, NY, USA, 1992.
- [NP99a] Moni Naor and Benny Pinkas. Oblivious Transfer and Polynomial Evaluation. In *Proceedings of STOC 1999*, pages 245–254. ACM Press, 1999.
- [NP99b] Moni Naor and Benny Pinkas. Oblivious Transfer with Adaptive Queries. In Michael J. Wiener, editor, *Proceedings of CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 573–590. Springer, 1999.
- [NS98] Phong Q. Nguyen and Jacques Stern. Cryptanalysis of the Ajtai-Dwork Cryptosystem. In Hugo Krawczyk, editor, *Proceedings of CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 223–242. Springer, 1998.
- [NY90] Moni Naor and Moti Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *Proceedings of the Twenty Second Annual ACM Symposium on Theory of Computing (STOC)*, pages 427–437. ACM Press, 1990.
- [Ore87] Yair Oren. On the Cunning Power of Cheating Verifiers: Some Observations about Zero Knowledge Proofs (Extended Abstract). In *Proceedings of FOCS 1987*, pages 462–471. IEEE Computer Society, 1987.
- [Pap93] Christos H. Papadimitriou. *Computational Complexity*. Addison Wesley, 1993.
- [Pas04] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In László Babai, editor, *Proceedings of STOC 2004*, pages 232–241. ACM Press, 2004.
- [Pin03] Benny Pinkas. Fair Secure Two-Party Computation. In Eli Biham, editor, *Proceedings of EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 87–105. Springer, 2003.
- [PR03] Rafael Pass and Alon Rosen. Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds. In *Proceedings of FOCS 2003*, pages 404–411. IEEE Computer Society, 2003.
- [PS04] Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In László Babai, editor, *Proceedings of STOC 2004*, pages 242–251. ACM Press, 2004.
- [PSW00] Birgit Pfitzmann, Matthias Schunter, and Michael Waidner. Secure Reactive Systems. RZ 3206 (#93252), IBM Research Division, ZÄijrich, May 2000.

- [PV06a] Sylvain Pasini and Serge Vaudenay. An Optimal Non-interactive Message Authentication Protocol. In David Pointcheval, editor, *Proceedings of CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 280–294. Springer, 2006.
- [PV06b] Sylvain Pasini and Serge Vaudenay. SAS-Based Authenticated Key Agreement. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Proceedings of PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 395–409. Springer, 2006.
- [PW00] Birgit Pfitzmann and Michael Waidner. Composition and integrity preservation of secure reactive systems. In *Proceedings of ACM CCS 2000*, pages 245–254, 2000.
- [Rog06] Phillip Rogaway. Formalizing Human Ignorance. In Phong Q. Nguyen, editor, *Proceedings of VIETCRYPT 2006*, volume 4341 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2006.
- [RS91] Charles Rackoff and Daniel R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In Joan Feigenbaum, editor, *Proceedings of CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 433–444. Springer, 1991.
- [RSA07] RSA Laboratories. The RSA Factoring Challenge RSA-640. Available from <http://www.rsa.com/rsalabs/>, 2007.
- [Sha49] Claude E. Shannon. Communication Theory of Secrecy Systems. *Bell Systems Technical Journal*, 28:656–715, 1949. Available as reprint from <http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf>.
- [Sha79] Adi Shamir. Factoring Numbers in  $O(\log n)$  Arithmetic Steps. *Information Processing Letters*, 8(1):28–31, 1979.
- [Sho97] Victor Shoup. Lower Bounds for Discrete Logarithms and Related Problems. In Walter Fumy, editor, *Proceedings of EUROCRYPT 1997*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266. Springer, 1997.
- [Sho04] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. Available from <http://eprint.iacr.org/>.
- [SJ00] Claus-Peter Schnorr and Markus Jakobsson. Security of Signed ElGamal Encryption. In Tatsuaki Okamoto, editor, *Proceedings of ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 73–89. Springer, 2000.
- [Smi94] Carl H. Smith. *A Recursive Introduction to the Theory of Computation*. Springer, 1994.

- [Sti86] Stephen M. Stigler. *The History of Statistics: The Measurement of Uncertainty before 1900*. Belknap Press of Harvard University Press, 1986.
- [Tur37] Alan. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42(2):230–265, 1937.
- [Vau05] Serge Vaudenay. Secure Communications over Insecure Channels Based on Short Authenticated Strings. In Victor Shoup, editor, *Proceedings of CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 309–326. Springer, 2005.
- [vL96] Michiel van Lambalgen. Randomness and Foundations of Probability: Von Mises’ Axiomatisation of Random Sequences. In T. S. Ferguson, L. S. Shapley, and J. B. MacQueen, editors, *Statistics, Probability and Game Theory: Papers in Honor of David Blackwell*, volume 30 of *Lecture Notes - Monograph Series*, pages 347–367, 1996.
- [WS04] David P. Woodruff and Jessica Staddon. Private inference control. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, editors, *Proceedings of ACM CCS 2004*, pages 188–197. ACM Press, 2004.
- [WW01] K. Wagner and G. Wechsung. *Computational Complexity*, volume 21 of *Mathematics and its applications*. Springer, 2001.
- [Yao82] Andrew Chi-Chih Yao. Theory and Applications of Trapdoor Functions (Extended Abstract). In *Proceedings of FOCS 1982*, pages 80–91. IEEE Computer Society, 1982.
- [Yao86] Andrew Chi-Chih Yao. How to Generate and Exchange Secrets (Extended Abstract). In *Proceedings of FOCS 1986*, pages 162–167. IEEE Computer Society, 1986.

# INDEX

- advantages
  - $\text{Adv}_G(\cdot)$ , 31, 78
  - $\text{Adv}_G^{\text{cr}}(\cdot)$ , 75
  - $\text{Adv}_G^{\text{ind}}(\cdot)$ , 23, 28, 78
  - $\text{Adv}_G^{\text{rom}}(\cdot)$ , 72
  - $\text{Adv}_G^{\text{std}}(\cdot)$ , 72
- adversarial distribution, 78
- adversaries, 28, 48–49, 90
  - adaptive adversary, 49
  - black-box adversary, 73
  - mobile adversary, 49
  - rational adversary, 34
  - relevant adversary, 72, 77
  - static adversary, 49
- adversary structure, 48, 102
- aggregate error, 23
- algorithms, 17–21
  - interactive algorithm, 19
  - polynomial-time algorithm, 13, 65
  - randomised algorithm, 19
  - $t$ -time algorithm, 20
- asymptotic complexity, 13, 64–68
  - $\text{negl}(\mathbb{k})$ , 13
  - $\text{poly}(\mathbb{k})$ , 13
  - $\text{polylog}(\mathbb{k})$ , 13
  - sub-exponential  $2^{o(\mathbb{k})}$ , 70
  - super-polynomial  $\mathbb{k}^{\omega(1)}$ , 70
- asymptotic security, 66, 68–71
- asymptotic security models
  - strong polynomial security, 68
  - strong statistical security, 71
  - weak polynomial security, 70
  - weak statistical security, 71
- asymptotically negligible, 13, 25, 66
- Bayesianism, 15
- belief, 16, 31, 50, 78–79, 80, 82
  - fundamental belief, 80, 82
- Bellare-Rogaway model, 113
- Big Oh notation, 13
- bootstrapping, 108, 113
- closeness, 53, 54, 83, 88
- coherent reasoning, 79–80
- collision resistance, 75
- common reference string, 74, 108, 113
- communication model, 19, 91
- computational context, 87
- computational devices
  - Random Access Machine, 18
  - Turing machine, 17
- computational distance, 24, 28, 78
- computational models
  - asynchronous model, 97
  - basic model, 28, 89
  - real-time model, 96
  - synchronous model, 96
- concurrent execution, 103, 107
- conditional probability, 15, 36–38
- construction family, 65
- correctness, 56, 58, 60, 112
- cryptosystems, 27
  - IND-CCA security, 27
  - IND-CPA security, 27, 79
- dedicated participants
  - administrator  $\mathcal{P}_{\text{na}}$ , 91
  - courier  $\mathcal{P}_{\text{nw}}$ , 91
  - scheduler  $\mathcal{P}_{\text{sc}}$ , 97
- distinguishers, see tests
- distributions, 12
  - adversarial distribution, 78
  - input distribution, 49
  - uniform distribution, 13
- encryption paradox, 75
- equivocability, 62
- exact security, 31–36, 38–40, 51–60, 87–89, 92–113
- external consistency, 46
- extractability, 62, 112
- fairness, 48
- fairness postulate, 77
- finite set policy, 7
- frequentism, 14
- games, 26–30
  - game chain, 38, 84
  - game description, 28–29
  - game tree, 40

- halting machine, 58
- hash function, 72, 74
- hashing paradox, 75
- horizon splitting, 39, 84
- hybrid argument, 84
- hypothesis testing, 22–26
  - aggregate error, 23
  - complex hypothesis, 22, 25
  - simple hypothesis, 22
- idealised functionality, 47–51
- independence, 15
- indistinguishability, 23, 27, 29, 61
- input distribution, 49
- input-privacy, 57
- interactive algorithm, 19
- interactive inference, *see* games
- interface, 104
- interpretations of probability, 14–15
  - objective probability, 14
  - subjective probability, 14
- isolated execution, 97–102
- Landau notation, 13
- malicious model, 56–60
  - correctness, 60
  - input-privacy, 57
  - output-consistency, 58
  - security, 59
- message authentication, 112
- negligible probability, 13, 24
- objective security, *see* exact security
- output-consistency, 58
- output-privacy, 56
- parallel execution, 101, 102
- Philosopher’s Stone, 95
- privacy, 55
- privacy loss, 51
- probability, 12–16
  - conditional probability, 15, 36
  - negligible probability, 13, 24
  - overwhelming probability, 13
- proof of knowledge, 69, 70
- proof system, 41–43
- PRP/PRF switching lemma, 37
- public inputs, 111
- public-key infrastructure, 75, 76, 111
- Random Access Machine, 18
- random oracle model, 72
- random tape model, 16
- random variable, 12
- randomised algorithm, 19
- rational adversary, 34
- rational entity, 79
- reduction schemata, 33, 83
  - $\mathcal{G}$ -SEC schema, 88
  - IND-CPA schema, 33
  - MPC-SEC schema, 53
  - MPC-SEC<sup>+</sup> schema, 94
  - UC-SEC schema, 103
- reductions, 32–36, 80–85
  - black-box reduction, 32, 61, 69
  - linear reduction, 35
  - quadratic reduction, 35
  - white-box reduction, 32, 61, 69, 85, 95
- relative advantage, 78
- relative security margin, 65
- relativised model, 43
- resilience principle, 4
- rewinding, 62
- risk profile, 50
- robust message scheduling, 94
- security parameter  $k$ , 13, 65
- security premise, 80, 82
- semantic security, 25, 29, 61
- semi-honest model, 55–56
  - correctness, 56
  - privacy, 55
- separations, 44
  - black-box separation, 44
  - oracle separation, 44
- sequential composability, 97–102
- shared setup, 108–111, 113
- simulatability, 60–63
  - black-box simulator, 61
  - white-box simulator, 61
- stand-alone security, 45–63, 92–95
  - security w.r.t. interface, 104
  - strong stand-alone security, 94
- statistical closeness, 54
- statistical distance, 16, 24, 28

- subjective equivalence assumption, 72
- subjective probability, 14, 50, 60, 78
- subjective security, 72–85, 110
- synchronisation errors, 93, 106, 113
  - escaping messages, 93
  - invading messages, 93
- tests, 22–24
  - computational test, 23
  - likelihood ratio test, 22
  - statistical test, 22
- tightness factor, 35
- time complexity, 19–21, 90
- time-success profile, 34
- time-success ratio, 34
- triangle inequality, 16, 24, 28, 39, 40
- trusted setup, 63, 74–77, 107–111
  - CRS model, 74, 108, 113
  - PKI model, 75, 111
- Turing machine, 17
- uniform distribution, 13
- universal composability, 102–113
- zero-knowledge proof, 59, 61, 69, 107



## **Publication P1**

Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On Private Scalar Product Computation for Privacy-Preserving Data Mining. Lecture Notes in Computer Science, volume 3506, pages 104–120. © Springer-Verlag. Reprinted with permission.



# On Private Scalar Product Computation for Privacy-Preserving Data Mining

Bart Goethals<sup>1</sup>, Sven Laur<sup>2</sup>, Helger Lipmaa<sup>2</sup>, and Taneli Mielikäinen<sup>1</sup>

<sup>1</sup> HIIT Basic Research Unit  
Department of Computer Science  
University of Helsinki, Finland  
{goethals,tmielika}@cs.helsinki.fi  
<sup>2</sup> Laboratory for Theoretical Computer Science  
Department of Computer Science and Engineering  
Helsinki University of Technology, Finland  
{slaur,helger}@tcs.hut.fi

**Abstract.** In mining and integrating data from multiple sources, there are many privacy and security issues. In several different contexts, the security of the full privacy-preserving data mining protocol depends on the security of the underlying private scalar product protocol. We show that two of the private scalar product protocols, one of which was proposed in a leading data mining conference, are insecure. We then describe a provably private scalar product protocol that is based on homomorphic encryption and improve its efficiency so that it can also be used on massive datasets.

**Keywords:** Privacy-preserving data mining, private scalar product protocol, vertically partitioned frequent pattern mining.

## 1 Introduction

Within the context of privacy-preserving data mining, several private (shared) scalar product protocols [DA01b,DA01a,DZ02,VC02] have been proposed. The goal is that one of the participants obtains the scalar product of the private vectors of all parties. Additionally, it is often required that no information about the private vectors, except what can be deduced from the scalar product, will be revealed during the protocol. Moreover, since data mining applications work with a huge amount of data, it is desirable that the scalar product protocol is also very efficient. A secure scalar product protocol has various applications in privacy-preserving data mining, starting with privacy-preserving frequent pattern mining on vertically distributed database [VC02] and ending with privacy-preserving cooperative statistical analysis [DA01a].

To give an idea of how such a protocol can be used, let us look at the protocol by Vaidya and Clifton for computing frequent itemsets from vertically partitioned transaction database [VC02]. A transaction database is a multi-set of subsets (*transactions*) of some finite set (of *items*). A transaction database

can be seen also as a binary matrix where each row corresponds to a transaction, each column corresponds to an item, and there is one in the entry  $(i, j)$  if and only if the transaction  $i$  contains the item  $j$ . An itemset is a subset of items. The frequency of an itemset in a transaction database is the fraction of transactions containing the itemset as their subset. (The support of an itemset is its frequency multiplied by the number of transactions in the database.) The  $\sigma$ -frequent itemsets (i.e., the frequent itemsets with minimum frequency threshold  $\sigma$ ) in a transaction database are the itemsets with frequency at least  $\sigma$ . Thus, mining the  $\sigma$ -frequent itemsets is equivalent to finding all subsets of columns of the binary matrix where at least a  $\sigma$ -fraction of rows have only ones in those columns. In a frequent itemset mining protocol for a vertically partitioned transaction database one party, Alice, has the projection of the database onto some items and another party, Bob, has the projection of database onto the rest of the items. The frequent itemset mining protocol of Vaidya and Clifton is based on the property that an itemset can be frequent only if all of its subsets are frequent. The candidate itemsets are generated and tested level-wise as in the APRIORI algorithm [AMS<sup>+</sup>96].

If an itemset contains items of only one party, then the party can compute the frequency privately and share it with the other parties without any additional privacy problems. The main challenge occurs when the support of a candidate itemset containing items from both parties needs to be computed. In that case, each party first computes which of the transactions contain the itemset within their own part of the database. This kind of information can be conveniently represented as binary vectors in which the  $i$ th entry represents whether or not the itemset is contained in the  $i$ th transaction. The number of transactions containing the itemset in the combined transaction database amounts to the scalar product between the corresponding binary vectors of Alice and Bob. A protocol, given by Vaidya and Clifton [VC02], attempts to compute the scalar product in a secure manner, by computing the scalar product on scrambled versions of the binary vectors, such that in the end of the protocol, both parties obtain the joint support without ever seeing each others vector. Their protocol reveals the supports of some infrequent itemsets, as not all candidate itemsets are frequent; this can be avoided by combining private shared scalar product protocols and Yao's circuits for frequency testing.

In this paper, we show that the private scalar product protocol of Vaidya and Clifton [VC02] is not private. Additionally, we are able to break another private (shared) scalar product protocol which was recently proposed by Du and Atallah [DA01a]. Our attacks against the Vaidya-Clifton and Du-Atallah protocols work in the simplest cryptographic model: namely, they enable one of the two parties to retrieve the private input of another party with probability, very close to 1, after the two parties have executed the corresponding protocol once.

While the attacks do not work for all possible private vectors of Alice and Bob, they show that before applying the Vaidya-Clifton and Du-Atallah protocols, one must carefully analyse whether it is safe to apply these protocols

in any concrete case. Moreover, the provided attacks can be readily generalised to work for a much larger fraction of private vectors in a more complex model where attack’s success probability does not have to be 1 (but just large enough for practical purposes, say 0.001) and/or when Alice and Bob re-execute the corresponding scalar product protocols from [DA01a,VC02] with similar private vectors. (Scalar product protocol from [DA01b] was recently analysed in [LL04].)

As a positive result, we describe a cryptographic protocol for computing scalar product. We prove that the new scalar product protocol is private—in a strong cryptographic sense—under standard cryptographic assumptions. More specifically, no probabilistic polynomial time algorithm substituting Alice (resp., Bob) can obtain a non-negligible amount of information about Bob’s (resp., Alice’s) private input, except what can be deduced from the private input and private output of Alice (resp., Bob). This means, in particular, that this protocol can be used a polynomial number of times (in the security parameter) with *any* private vectors of Alice and Bob in *any* context. In practice, the latter means “an arbitrary number of times”. Finally, we show that by using some optimisation tricks, the proposed protocol can be made very efficient: we show how to separately optimise for Alice’s and Bob’s computation, and for the communication of the new protocol. In particular, the communication-optimal version is more communication-efficient than either of the Vaidya-Clifton or the Du-Atallah protocols.

**Road-map.** In Section 2, we describe the necessary cryptographic preliminaries. In Section 3, we analyse some previous private scalar product protocols. In Section 4, we propose a new scalar product protocol, prove its security and propose some important optimisations. We finish with conclusions and acknowledgements.

## 2 Cryptographic Preliminaries

**Secure Multi-Party and Two-Party Computation.** To guarantee that a protocol is secure in as many applications as possible, one should use the secure multi-party and two-party techniques [Gol04]. Briefly, a two-party protocol between Alice and Bob is *secure* when privacy and correctness are guaranteed for both Alice and Bob. It is said that a protocol *protects privacy*, when the information that is leaked by the distributed computation is limited to the information that can be learned from the designated output of the computation [Pin02].

There are several different security models where one can prove the security of a protocol in. The simplest setting is the *semi-honest* model, where it is assumed that both Alice and Bob follow the protocol, but they are also curious: that is, they store all exchanged data and try to deduce information from it. In the *malicious model*, no assumption is made about the behaviour of Alice and Bob, and it is required that the privacy of one party is preserved even in the case of an arbitrary behaviour of the second party. Most of the papers on privacy-preserving data mining provide only security in the semi-honest model.

Such a protocol can be made secure in the malicious model when accompanied with zero-knowledge proofs that both parties follow the protocol. However, such proofs are usually too inefficient to be used in data mining applications.

**Homomorphic public-key cryptosystems.** A public-key cryptosystem  $\Pi$  is a triple  $(\text{Gen}, \text{Enc}, \text{Dec})$  of probabilistic polynomial-time algorithms for key-generation, encryption and decryption. The security of a public-key cryptosystem is determined by a security parameter  $k$ . For a fixed  $k$ , it should take more than polynomial in  $k$  operations to break the cryptosystem. Together with increased security, larger  $k$  means also larger keys and ciphertexts. The key generation algorithm generates, on input  $1^k = 1 \dots 1$  ( $k$  ones) a valid pair  $(\text{sk}, \text{pk})$  of private and public keys that corresponds to the security parameter  $k$ . For a fixed key pair  $(\text{sk}, \text{pk})$ , let  $P(\text{sk})$  denote the plaintext space of  $\Pi$ . The encryption algorithm  $\text{Enc}$  takes as an input a plaintext  $m \in P(\text{sk})$ , a random value  $r$  and a public key  $\text{pk}$  and outputs the corresponding ciphertext  $\text{Enc}_{\text{pk}}(m; r)$ . The decryption algorithm  $\text{Dec}$  takes as an input a ciphertext  $c$  and a private key  $\text{sk}$  (corresponding to the public key  $\text{pk}$ ) and outputs a plaintext  $\text{Dec}_{\text{sk}}(c)$ . It is required that  $\text{Dec}_{\text{sk}}(\text{Enc}_{\text{pk}}(m; r)) = m$  for any  $m \in P(\text{sk})$ ,  $\text{pk}$  and  $r$ .

A public-key cryptosystem is *semantically secure* (IND-CPA secure) when a probabilistic polynomial-time adversary cannot distinguish between random encryptions of two elements, chosen by herself. We denote the encryption of a message  $m$  by  $\text{Enc}_{\text{pk}}(m; r)$ , where  $\text{pk}$  is the corresponding public key and  $r$  is the used random string. A public-key cryptosystem is *homomorphic* when  $\text{Enc}_{\text{pk}}(m_1; r_1) \cdot \text{Enc}_{\text{pk}}(m_2; r_2) = \text{Enc}_{\text{pk}}(m_1 + m_2; r_1 \cdot r_2)$ , where  $+$  is a group operation and  $\cdot$  is a groupoid operation. This means that a party can add encrypted plaintexts by doing simple computations with ciphertexts, *without* having the secret key. Usually,  $P(\text{sk}) = \mathbb{Z}_m$  for some large  $m$ . One of the most efficient currently known semantically secure homomorphic cryptosystems was proposed by Paillier cryptosystem [Pai99] and then improved by Damgård and Jurik [DJ01]. In Paillier's case,  $P(\text{sk}) = \mathbb{Z}_m$  with  $m \geq 2^{1024}$ . One can effectively assume that  $m$  is as large as say  $2^{4096}$ , when using the Damgård-Jurik cryptosystem [DJ01]. We will assume that  $k$  is the bit length of the plaintexts, thus  $k \geq 1024$ .

**Oblivious transfer.** In an  $\binom{n}{1}$ -oblivious transfer protocol, Bob has a database  $(\mathcal{D}_1, \dots, \mathcal{D}_n)$  and Alice has an index  $i \in [n]$ . The goal is for Alice to retrieve the element  $\mathcal{D}_i$  without revealing her index  $i$  to Bob, and Bob does not want Alice to get to know anything about the other elements in his database apart from the element she asks for. Recently, Lipmaa [Lip04] proposed an asymptotically efficient  $\binom{n}{1}$ -oblivious transfer protocol with communication  $\Theta(\log^2 n)k$ .

### 3 Cryptanalysis of Proposed Private SP Protocols

Before cryptanalysing some of the previously proposed private scalar product and private shared scalar product protocols, we must define what does it mean

to attack one. Next, we will give a somewhat intuitive definition. For simplicity, we will require that all arithmetic is done in  $\mathbb{Z}_m$  for some  $m$ .

We call a protocol between Alice and Bob a *scalar product* (SP) protocol when Bob obtains, on Alice’s private input  $\mathbf{x} = (x_1, \dots, x_N) \in \mathbb{Z}_m^N$  and on Bob’s private input  $\mathbf{y} = (y_1, \dots, y_N) \in \mathbb{Z}_m^N$ , the scalar product  $\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^N x_i y_i$ . A protocol is a *shared scalar product (SSP) protocol* when Alice receives a uniformly distributed random value  $s_A \in \mathbb{Z}_m$  and Bob receives a dependent uniformly distributed random value  $s_B \in \mathbb{Z}_m$ , such that  $s_A + s_B \equiv \mathbf{x} \cdot \mathbf{y} \pmod{m}$ . A scalar product protocol is *private* when after executing the protocol, Bob obtains no more knowledge than  $\mathbf{x} \cdot \mathbf{y}$  and Alice obtains no new knowledge at all. In particular, Alice gets to know nothing new about Bob’s vector and Bob gets to know nothing about Alice’s vector that is not implied by  $\mathbf{x}$  and  $\mathbf{x} \cdot \mathbf{y}$ . A *private shared scalar product protocol* is defined analogously.

Recently, several researchers from the data mining community have proposed private SSP and SP protocols [DA01b, DA01a, DZ02, VC02], that were primarily meant to be used in the context of privacy-preserving data mining. Most of the proposed solutions try to achieve information-theoretical security—that is, without relying on any computational assumption—by using additive or linear noise to mask the values. In almost all such solutions, one can construct a system of linear equations based on the specification of the protocol, and solve it for the secret values. We will next demonstrate that explicitly in the case of the protocols from [DA01a, VC02].

### 3.1 Vaidya-Clifton Private Scalar Product Protocol

First, we analyse the Vaidya-Clifton private SP protocol [VC02], depicted by Protocol 1. For the sake of simplicity, we assume that the database size is  $N = \ell n$ , where  $n$  is a block size and  $\ell$  is the number of blocks. We represent each  $N$ -dimensional vector  $\mathbf{z}$  either as  $\mathbf{z} = (z_1, \dots, z_N)$  or  $\mathbf{z} = (\mathbf{z}[\mathbf{1}], \dots, \mathbf{z}[\ell])$ , where  $\mathbf{z}[\mathbf{i}] = (z_{(i-1)n+1}, \dots, z_{in})$ . We denote the  $n$ -dimensional vectors  $(1, \dots, 1)$  and  $(0, \dots, 0)$  by  $\mathbf{1}$  and  $\mathbf{0}$ .

Protocol 1 is a slight modification of the original Vaidya-Clifton protocol. Namely, in the original protocol all scalars belong to  $\mathbb{R}$ , while in Protocol 1 they belong to  $\mathbb{Z}_m$  with  $m > N$ . Our modifications make the protocol more applicable and also more secure for the next reasons. First, as computers can use only limited precision, there will be stability and correctness problems when computing over real numbers. Second, adding random noise  $r$  from  $\mathbb{R}$  to value  $x$  from  $\mathbb{R}$  does not perfectly hide  $x$  since it is impossible to choose  $r$  uniformly at random from  $\mathbb{R}$ , or even from  $\mathbb{N}$ . Therefore, cryptanalysis of the original Vaidya-Clifton protocol is simpler and attacks against it are more dangerous when we consider their protocol as working in  $\mathbb{R}$ .

In the following, we explicitly assume that  $m$  is prime. Proposed attacks also work with composite  $m$ , but then one would have to tackle many insubstantial yet technical details. We will also establish some additional notation. First, for any  $\mathcal{I} = \{i_1, \dots, i_j\} \subseteq [N]$  with  $|\mathcal{I}| = j$ , any vector  $\mathbf{x}$  and any matrix  $M$ , let  $\mathbf{x}_{\mathcal{I}} = (x_{i_1}, \dots, x_{i_j})$  and  $M_{\mathcal{I}}$  denote the sub-matrix of  $M$  that consists of the

PRIVATE INPUT OF ALICE:  $\mathbf{x} \in \{0, 1\}^N$   
PRIVATE INPUT OF BOB:  $\mathbf{y} \in \{0, 1\}^N$   
PRIVATE OUTPUT OF BOB: Scalar product  $\mathbf{x} \cdot \mathbf{y} \pmod m$ .

1. Alice and Bob jointly do:  
Generate a random invertible  $N \times N$  matrix  $C$ .
2. Alice does:  
Generate a random vector  $\mathbf{p} \in \mathbb{Z}_m^N$ .  
Send  $\mathbf{u} \leftarrow \mathbf{x} + C\mathbf{p}$  to Bob.
3. Bob does:  
Generate  $\ell$  random values  $s_1, \dots, s_\ell \in \mathbb{Z}_m$ .  
Send  $\mathbf{v} \leftarrow C^T \mathbf{y} + \mathbf{r}$ , where  $\mathbf{r}[i] \leftarrow s_i \mathbf{1}$ , to Alice.
4. Alice does:  
Set  $t_0 := \mathbf{v} \cdot \mathbf{p}$ .  
For  $i \in \{1, \dots, \ell\}$ , set  $t_i := \sum_{j=1}^n p[j] r_j$ .  
Send  $(t_0, t_1, \dots, t_\ell)$  to Bob.
5. Bob does:  
Return  $\mathbf{u} \cdot \mathbf{y} - t_0 + \sum_{i=1}^{\ell} s_i t_i$ .

**Protocol 1:** Vaidya-Clifton private shared scalar product protocol. (All computations are done modulo a public  $m$ .)

rows  $\mathcal{I} = \{i_1, \dots, i_j\}$ . Second,  $C$  is invertible and known to both Alice and Bob. Therefore, define  $\mathbf{a}_i := (C^T)^{-1} \mathbf{e}_i \pmod m$ , where  $\mathbf{e}_i[j] = \mathbf{1}$  if  $i = j$  and  $\mathbf{e}_i[j] = \mathbf{0}$ , otherwise. Define  $\boldsymbol{\omega} := (C^T)^{-1} \mathbf{v}$ . Then  $(C^T)^{-1} \mathbf{r} \equiv (C^T)^{-1} (s_1 \mathbf{1}, \dots, s_\ell \mathbf{1}) \equiv \sum_{i=1}^{\ell} s_i \mathbf{a}_i \pmod m$ ,  $\boldsymbol{\omega} \equiv \mathbf{y} + \sum_{i=1}^{\ell} s_i \mathbf{a}_i \pmod m$  and  $t_i \equiv \mathbf{e}_i \cdot \mathbf{p} \equiv \mathbf{a}_i \cdot C\mathbf{p} \pmod m$  for  $i \geq 1$ .

First, we show that if the vector  $\mathbf{y}$  has a low support then Alice is guaranteed to learn half coefficients  $y_i$ —and with a high probability the whole vector  $\mathbf{y}$ —after just executing Protocol 1 once.

**Lemma 1.** *As previously, let  $\text{supp}(\mathbf{y}) := |\{y : y_i \neq 0\}|$  be the support of  $\mathbf{y}$ . Assume that  $N \geq (2 \text{supp}(\mathbf{y}) + 1)\ell$ . After just executing Protocol 1 once, a semi-honest Alice obtains at least half of the coefficients of  $\mathbf{y}$ , with probability 1, by solving  $2 \text{supp}(\mathbf{y}) + 1$  systems of linear equations in  $\ell$  variables.*

*Proof.* Let  $M$  be the matrix with column vectors  $\mathbf{a}_1, \dots, \mathbf{a}_\ell$ . Let  $\mathbf{s} = (s_1, \dots, s_\ell)$ . The attack is based on the observation that the equality  $M\mathbf{s} \equiv \boldsymbol{\omega} - \mathbf{y} \pmod m$  gives Alice a system of  $N$  linear equations in  $\ell$  unknowns  $s_j$ . The values  $v_i$  and vectors  $\mathbf{a}_1, \dots, \mathbf{a}_\ell$  are known to Alice; the values  $y_i \in \{0, 1\}$  are unknown. Alice partitions the set  $[N]$  iteratively into  $\geq N/\ell$  (non-empty) parts  $\mathcal{I}_k$  as follows: Denote  $\mathcal{J}_k := [N] \setminus \bigcup_{i < k} \mathcal{I}_i$ . Alice chooses an  $\mathcal{I}_k \subseteq \mathcal{J}_k$ , such that the matrix  $M_{\mathcal{I}_k}$  has the maximal possible rank with respect to  $\mathcal{J}_k$  and  $\mathcal{I}_k$  is minimal unless the rank of  $M_{\mathcal{J}_k}$  is zero. In particular,  $M_{\mathcal{J}_k} = D_k M_{\mathcal{I}_k}$  for some matrix  $D_k$ . If rank of  $M_{\mathcal{J}_k}$  is zero then Alice chooses a random index from  $\mathcal{J}_k$ . Note that  $M_{\mathcal{J}_k} = D_k M_{\mathcal{I}_k}$  still holds for an appropriate zero matrix  $D_k$ .

Now, there are at least  $N/\ell \geq 2 \text{supp}(\mathbf{y}) + 1$  parts  $\mathcal{I}_k$ . For a majority of indices  $k$  (we say that such indices  $k$  are “good”),  $\mathbf{y}_{\mathcal{I}_k}$  is a zero vector. Therefore, in the majority of the cases, Alice obtains the correct values  $\mathbf{s}_{\mathcal{I}_k}$  by solving the equation  $M_{\mathcal{I}_k} \mathbf{s} = \boldsymbol{\omega}_{\mathcal{I}_k}$ . Since  $M_{\mathcal{J}_k} \mathbf{s} = D_k M_{\mathcal{I}_k} \mathbf{s}$ , the value of  $\mathbf{y}_{\mathcal{J}_k}$  is uniquely determined by  $\mathbf{s}_{\mathcal{I}_k}$ . Moreover, the smallest “good”  $k = k_0$  satisfies  $k_0 \leq \text{supp}(\mathbf{y}) + 1$ . The solution  $\mathbf{s}$  of  $M_{\mathcal{I}_{k_0}} \mathbf{s} = (\boldsymbol{\omega})_{\mathcal{I}_{k_0}}$  is consistent with the solutions that correspond to other “good”  $k$ ’s, that is,  $M_{\mathcal{I}_k} \cdot \mathbf{s}_{\mathcal{I}_{k_0}} = \boldsymbol{\omega}_{\mathcal{I}_k}$  for all “good” indices  $k > k_0$ . Therefore, Alice can find all “good” indices  $k$  by majority voting. She also obtains all coordinates of  $\mathbf{y}_{\mathcal{J}_{k_0}}$ .  $\square$

If  $|\mathcal{I}_{k_0}| = \ell$  then all coordinates of  $\mathbf{y}$  are revealed, otherwise coefficients are revealed for all sets  $|\mathcal{I}_k| \leq |\mathcal{I}_{k_0}|$ , as any solution to  $M_{\mathcal{I}_{k_0}} \mathbf{s} = \boldsymbol{\omega}_{\mathcal{I}_{k_0}}$  uniquely determines  $\mathbf{y}_{\mathcal{J}_{k_0}} = \boldsymbol{\omega}_{\mathcal{J}_{k_0}} - D_{k_0} \boldsymbol{\omega}_{\mathcal{I}_{k_0}}$ . The next result shows that  $\mathbf{y}$  is revealed almost certainly.

**Lemma 2.** *Let  $\mathcal{I}_k$  be defined as in the proof of the previous lemma. Then  $\Pr[|\mathcal{I}_k| = |\mathcal{I}_{k+1}|] = \prod_{i=0}^{d-1} (1 - m^{-|\mathcal{J}_k|+i})$ . Thus, the probability that all coefficients are revealed is approximately  $(1 - m^{-N/2})^{\text{supp}(\mathbf{y})\ell} \approx 1 - \text{supp}(\mathbf{y})\ell m^{-N/2}$ .*

*Proof.* Consider all possible vector assignments of  $\mathbf{a}_1, \dots, \mathbf{a}_\ell$  that are consistent with the choice of  $\mathcal{I}_1, \dots, \mathcal{I}_k$ ; that is, such assignments, for which  $M_{\mathcal{J}_k} = D'_k M_{\mathcal{I}_k}$  for some  $D'_k$ . The latter is equivalent to the assumption that rows of  $M_{\mathcal{J}_k}$  are randomly sampled from a vector space of dimension  $|\mathcal{I}_k|$ . By a standard result [vLW92, p. 303], the probability that  $\text{rank}(M_{\mathcal{J}_k}) = |\mathcal{I}_k|$  is equal to  $\prod_{i=0}^{|\mathcal{I}_k|-1} (1 - m^{-|\mathcal{J}_k|+i})$ . Hence, the first claim is proven. Now,  $\mathbf{y}$  is completely determined if  $|\mathcal{I}_{\text{supp}(\mathbf{y})+1}| = \ell$ . As  $|\mathcal{I}_1| = \ell$  by the protocol construction and for  $k < \text{supp}(\mathbf{y})$ ,  $|\mathcal{J}_{\text{supp}(\mathbf{y})}| > N/2$ , the second claim follows from a straightforward calculation.  $\square$

If we give more power to Alice, she will be able to do much better. Assume that Protocol 1 is run twice with the same input vector  $\mathbf{y}$ ; let  $\mathbf{a}_1, \dots, \mathbf{a}_\ell$  and  $\mathbf{a}'_1, \dots, \mathbf{a}'_\ell$  be vectors, computed from the random matrices  $C$  and  $C'$  as previously. Then,  $\boldsymbol{\omega} - \boldsymbol{\omega}' = \sum_{i=1}^{\ell} s_i \mathbf{a}_i - \sum_{i=1}^{\ell} s'_i \mathbf{a}'_i$ . With high probability, this determines  $\mathbf{s}$  and  $\mathbf{s}'$  uniquely. To avoid similar attacks, Bob must never run Protocol 1 twice with the same input  $\mathbf{y}$  but different matrices  $C$ . The next lemma shows that also Alice must never run Protocol 1 twice with the same input  $\mathbf{x}$  but different matrices  $C$ .

**Lemma 3.** *If Protocol 1 is re-executed  $k > N/\ell$  times with the same  $\mathbf{x}$ , Bob obtains  $\mathbf{x}$  with probability higher than  $\prod_{i=0}^{N-1} (1 - m^{-k\ell+i})$ .*

*Proof.* Each execution of Protocol 1 provides  $\ell$  linear equations  $\mathbf{a}_i \cdot \mathbf{u} = \mathbf{a}_i \cdot \mathbf{x} + \mathbf{a}_i \cdot C \mathbf{p} = \mathbf{a}_i \cdot \mathbf{x} + t_i$  for  $i \in \{1, \dots, \ell\}$ . As  $\mathbf{a}_1, \dots, \mathbf{a}_\ell$  are chosen randomly, similar argumentation as in Lemma 2 gives the probability estimate.  $\square$

Finally, we get another efficient attack when we consider itemsets with almost the same support. For example, assume that Alice knows that  $\text{supp}(\mathbf{y} - \mathbf{y}') <$

$N/(4\ell) - 1/2$ . Then, by using Lemma 1, Alice can determine  $\mathbf{s}$  and  $\mathbf{s}'$  from the equation  $\boldsymbol{\omega} - \boldsymbol{\omega}' = \mathbf{y} - \mathbf{y}' + \sum_{i=1}^{\ell} s_i \mathbf{a}_i - \sum_{i=1}^{\ell} s'_i \mathbf{a}'_i$ ; therefore, she obtains  $\mathbf{y}$  and  $\mathbf{y}'$ . This attack works with any choice of  $C$ . The condition  $\text{supp}(\mathbf{y} - \mathbf{y}') \ll N$  is not so rare in the context of frequent itemset mining. Moreover, several optimisations of APRIORI are devised to exploit such shortcuts. To analyse the applicability of low support attacks, we need additional notations. Let  $\text{supp}(I)$  denote the support of the itemset  $I$  and  $\mathbf{y}_I$  the corresponding vector, i.e.  $y_{I,k} = 1$  iff the  $k$ th row contains items  $I$ . We say that  $I$  is a closed frequent itemset, iff  $\text{supp}(I)$  is over frequency threshold and for any proper superset  $J \supsetneq I$ ,  $\text{supp}(I) > \text{supp}(J)$ . Now, if the frequent itemset  $I$  is not closed, then the APRIORI algorithm discovers  $J \supset I$  such that  $\text{supp}(\mathbf{y}_I - \mathbf{y}_J) = 0$  and Alice can apply the attack. The ratio  $\rho$  between frequent and frequent closed sets describes the average number of vectors revealed by a single closed set. Empirical results [PHM00] on standard data mining benchmarks indicate that  $\rho$  can range from 2 to 100 depending on the frequency threshold, when the database contains some highly correlated items.

The analysis can be extended further by using notion of frequent  $\delta$ -free sets. A itemset  $I$  is  $\delta$ -free if and only if for any proper subset  $J$  of  $I$ ,  $\text{supp}(\mathbf{y}_J - \mathbf{y}_I) > \delta$ . In other words, an itemset  $I$  is not  $\delta$ -free if and only if there is  $J \subsetneq I$  with  $\text{supp}(\mathbf{y}_J - \mathbf{y}_I) \leq \delta$ . Again, empirical results [BBR03, BB00] on standard data mining benchmarks show that the number of frequent  $\delta$ -free sets with  $\delta \in [0, 20]$  is several magnitudes smaller than the number of frequent sets, when database contain highly correlated items. To conclude, low support differences are quite common for many practical data sets and thus the Vaidya-Clifton scalar product protocol is insecure for frequent itemset mining.

**Remark on [VC02, Section 5.2].** In [VC02, Section 5.2], Vaidya and Clifton note that the fact that  $x_i$  and  $y_i$  belong to  $\{0, 1\}$  can create a disclosure risk. They propose two solutions. The first consists of “*cleverly*” selecting the matrix  $C$  so that it is not evident which of the values of  $x_i$  and  $y_i$  are 1’s. Lemma 1 states that such a “*clever*” choice is impossible in general since at least a half of  $\mathbf{y}$ ’s coordinates is revealed for every matrix  $C$ . Besides, the solution is not fully spelled out and no security proofs are given. Another solution from [VC02, Section 5.2] is said to increase the security of Bob but decrease the security of Alice, but again, no security proofs are given. Thus, it is difficult to estimate the exact security of the proposed solutions. It seems that neither of these mentioned solutions is secure against our attacks.

**Communication and computation of Vaidya-Clifton protocol.** Alice and Bob must both know  $C$ , thus the communication of the Vaidya-Clifton protocol is approximately  $N^2 \log m$  bits. In the version of the scalar product protocol where no privacy is guaranteed, Alice just sends her vector ( $N$  bits) to Bob, who returns the scalar product ( $\lceil \log_2 N \rceil$  bits). Define the communication overhead of a private scalar protocol  $P$  to be equal to  $C(P)/N$ , where  $C(P)$  is the number of bits communicated in the protocol  $P$ . Thus, the communication overhead of

PRIVATE INPUTS: Vectors  $\mathbf{x} \in \{0, 1\}^N$  and  $\mathbf{y} \in \{0, 1\}^N$ .  
PRIVATE OUTPUTS: Shares  $s_A + s_B \equiv \mathbf{x} \cdot \mathbf{y} \pmod{m}$ .

1. Alice does:
  - Generate random  $\mathbf{v}_1, \dots, \mathbf{v}_{d-1} \leftarrow \mathbb{Z}_m^N$ .
  - Set  $\mathbf{v}_d := \mathbf{x} - \sum_{i=1}^{d-1} \mathbf{v}_i$  and  $s_A := 0$ .
2. For  $i = 1$  to  $d$  do
  - (a) Alice does:
    - Generate random  $\ell_i \in \{1, \dots, p\}$ .
    - Set  $\mathbf{h}_{i\ell_i} := \mathbf{v}_i$ .
    - For  $j \in \{1, \dots, \ell_i - 1, \ell_i + 1, \dots, p\}$ : Generate random  $\mathbf{h}_{ij} \in \mathbb{Z}_m^n$ .
    - Send  $(\mathbf{h}_{i1}, \dots, \mathbf{h}_{ip})$  to Bob.
  - (b) Bob does:
    - Generate random  $r_i \in \mathbb{Z}_m$ .
    - For  $j \in \{1, \dots, p\}$ : Set  $z_{ij} := \mathbf{h}_{ij} \cdot \mathbf{y} + r_i$ .
  - (c) Alice does:
    - Use  $\binom{p}{1}$ -oblivious transfer to retrieve  $z_{i\ell_i}$  from  $(z_{i1}, \dots, z_{ip})$ .
    - Set  $s_A := s_A + z_{i\ell_i}$ .
3. Alice outputs  $s_A$ , Bob outputs  $s_B = -\sum_{i=1}^d r_i$ .

**Protocol 2:** Du-Atallah private SSP protocol. Here,  $m > N$  is a public modulus

the Vaidya-Clifton private SP protocol is  $Nm$ . Computation is dominated by  $\Theta(N^2)$  multiplications and additions in  $\mathbb{Z}_m$ . The new scalar product protocol, that we will propose in this paper, is both more secure and more efficient.

### 3.2 Du-Atallah Private Scalar Product Protocol

Du and Atallah proposed another private SSP protocol [DA01a], depicted by Protocol 2. We show that also this protocol cannot handle binary vectors with low support.

Since Protocol 2 chooses the values  $r_i$  randomly,  $s_A$  is a random value and therefore Alice does not learn anything about  $\mathbf{y}$ . To learn  $\mathbf{x}$ , Bob must guess correctly the values  $\ell_i$  for all  $i$ . Since the probability of a random guess is  $p^{-d}$ , Du and Atallah argue that this protocol is secure when  $p^d > 2^{80}$ . Bob can do much better, however.

**Lemma 4.** *Assume  $N \geq (2 \text{supp}(\mathbf{x}) + 1)pd$ . Then, with probability 1, Bob finds at least  $N/2$  coordinates of  $\mathbf{x}$  by solving  $\text{supp}(\mathbf{x}) + 1$  systems of linear equations, each having dimension  $pd - 1$ . With high probability  $\approx (1 - m^{-N/2})^{\text{supp}(\mathbf{x})(pd-1)} \approx 1 - \text{supp}(\mathbf{x})(pd - 1)m^{-N/2}$ , Bob obtains the whole vector  $\mathbf{x}$ .*

*Proof.* Bob knows that  $\sum_{i=1}^d \mathbf{h}_{ij_i} = \mathbf{x}$  for some values  $j_i$ . Equivalently,

$$\sum_{i=1}^d \sum_{j=1}^p c_{ij} \mathbf{h}_{ij} = \mathbf{x} ,$$

where  $c_{ij} = 1$  if  $j = j_i$  and  $c_{ij} = 0$ , otherwise. Exactly as Alice did in the proof of Lemma 1, Bob iteratively partitions  $[N]$  into subsets  $\mathcal{I}_k$  with maximal possible rank. Hence, a solution to  $\sum_{i,j} c_{ij}(\mathbf{h}_{ij})_{\mathcal{I}_{k_0}} = \mathbf{0}$  uniquely determines  $\mathbf{x}_{\mathcal{I}_k} = \sum_{i,j} c_{ij}(\mathbf{h}_{ij})_{\mathcal{I}_k}$  for  $k > k_0$ . On the other hand, Bob creates at least  $2 \text{supp}(\mathbf{x}) + 1$  partitions  $\mathcal{I}_k$ . Thus, there exists a  $k \leq \text{supp}(\mathbf{x}) + 1$ , such that  $\mathbf{x}_{\mathcal{I}_k} = \mathbf{0}$ . As in the proof of Lemma 1, we can determine the first “good”  $k_0 \leq \text{supp}(\mathbf{x}) + 1$  by using majority voting.

To reduce the amount of computations, Bob can ignore all sets  $|\mathcal{I}_k| = pd$ . For any “good”  $k$ ,  $|\mathcal{I}_k| \leq pd - 1$ , as  $\mathbf{x}_{\mathcal{I}_k} = \mathbf{0}$  and the homogeneous system  $\sum_{i,j} c_{ij}(\mathbf{h}_{ij})_{\mathcal{I}_k} = \mathbf{0}$  has a nontrivial solution.

The proof of the second claim is similar to the proof of Lemma 2, since it is sufficient that  $pd - 1$  random vectors are linearly independent, and  $|\mathcal{I}_1| \geq pd - 1$  by construction.  $\square$

This protocol has another serious weakness, since with high probability slightly more than  $pd$  coordinates of  $\mathbf{x}$  allow to determine correct  $c_{ij}$  and thus also reveal other coordinates. Therefore, a leakage of  $pd$  database entries, can reveal the whole vector (database) and thus  $pd$  must be large, say more than 200. On the other hand, this protocol is very inefficient when  $pd$  is large.

**Communication and computation complexity.** Assume  $p^d > 2^{80}$ . Then the communication of the Du-Atallah private SSP protocol is  $dpN + dt_p$ , where  $t_p$  is the communication complexity of the  $\binom{p}{1}$ -oblivious transfer protocol. This is minimal when  $d$  is maximised, i.e., when  $p = 2$ . Taking the efficient  $\binom{p}{1}$ -oblivious transfer protocol from [AIR01], one has  $t_2 = 3k$ , where  $k \approx 1024$  is the security parameter. Then the communication is  $2dN + 3dk$  bits for  $d \geq 80$  and  $k \geq 1024$ . Taking  $d = 80$  and  $k = 1024$ , we get communication  $160N + 245760$  bits. However, Lemma 4 indicates that for the security of the Du-Atallah protocol, one should pick  $p$  and  $d$  such that  $pd$  is quite large. For example, picking  $p = 2^{11}$  and  $d = 8$  might result in an acceptable security level, but then the communication of the protocol will be  $2^{14} \cdot N + dt_p$  bits.

## 4 Cryptographic Private SSP Protocol

In this section we describe a private SSP protocol (Protocol 3) that is based on homomorphic encryption. Note that a private SP protocol can be obtained from it by defining  $s_B \leftarrow 0$ .

**Theorem 1.** *Assume that  $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$  is a semantically secure homomorphic public-key cryptosystem with  $P(\text{sk}) = \mathbb{Z}_m$  for some large  $m$ . Set  $\mu := \lfloor \sqrt{m/N} \rfloor$ . Protocol 3 is a secure SSP protocol in the semi-honest model, assuming that  $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_\mu^N$ . Alice’s privacy is guaranteed when Bob is a probabilistic polynomial-time machine. Bob’s privacy is information-theoretical.*

*Proof.* Clearly, the protocol is correct if the participants are honest. Since the cryptosystem is semantically secure, Bob only sees  $N$  random ciphertexts, for

PRIVATE INPUTS: Private vectors  $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_\mu^N$ .  
PRIVATE OUTPUTS: Shares  $s_A + s_B \equiv \mathbf{x} \cdot \mathbf{y} \pmod m$

1. Setup phase. Alice does:
  - Generate a private and public key pair  $(\text{sk}, \text{pk})$ .
  - Send  $\text{pk}$  to Bob.
2. Alice does for  $i \in \{1, \dots, N\}$ :
  - Generate a random new string  $r_i$ .
  - Send  $c_i = \text{Enc}_{\text{pk}}(x_i; r_i)$  to Bob.
3. Bob does:
  - Set  $w \leftarrow \prod_{i=1}^N c_i^{y_i}$ .
  - Generate a random plaintext  $s_B$  and a random nonce  $r'$ .
  - Send  $w' = w \cdot \text{Enc}_{\text{pk}}(-s_B; r')$  to Alice.
4. Alice does: Compute  $s_A = \text{Dec}_{\text{sk}}(w') = \mathbf{x} \cdot \mathbf{y} - s_B$ .

**Protocol 3:** Private homomorphic SSP protocol

which he cannot guess the plaintexts. In particular, this holds even when Bob has given two candidate vectors  $\mathbf{x}_1$  and  $\mathbf{x}_2$  to Alice and Alice has randomly chosen one of them,  $\mathbf{x} := \mathbf{x}_b$ . Even after a polynomial number of protocol executions with Alice’s input, Bob will gain only an insignificant amount of information about  $\mathbf{x}_b$  that will not help him in guessing the value of  $b$ . (This roughly corresponds to the standard notion of semantic security.) On the other hand, Alice only sees a random encryption of  $s_A = \mathbf{x} \cdot \mathbf{y} - s_B$ , where  $s_B$  is random. But Alice has the key anyways, so she can decrypt this message. Thus, Alice obtains no information at all.  $\square$

(Note that if  $m > 2^{1024}$  and  $N \approx 2^{16}$  then  $\mu \geq 2^{504}$ .) In Appendix A, we describe an extension of this protocol to more than two parties.

**Practical considerations.** Note that when Alice and Bob need to execute this protocol several times, they can reuse public and private keys and thus the setup phase can be executed only once. Public key cryptography is computationally demanding. To estimate the computational cost of the new scalar product protocol, we must count encryptions, decryptions and multiplications of ciphertexts. Bob must perform  $N$  exponentiations and 1 encryption. Alice has to perform  $N$  encryptions and 1 decryption.

In the specifically interesting case when  $x_i, y_i \in \{0, 1\}$  (e.g., when  $\mathbf{x}$  and  $\mathbf{y}$  correspond to characteristic functions of two sets  $X$  and  $Y$ ; then  $\mathbf{x} \cdot \mathbf{y} = |X \cap Y|$ ), this protocol can be further optimised. Namely, Alice can pre-compute and then store a large table of random encryptions of 0’s and 1’s. Then every “encryption” just corresponds of fetching a new element from the correct table; this can be done very quickly. Bob has to perform 1 encryption and  $\text{supp}(\mathbf{y})$  multiplications, since the exponents  $y_i$  are all Boolean. (When  $y_i = 0$  then  $c_i^{y_i} = 1$  and otherwise  $c_i^{y_i} = c_i$ .)

The current hardware allows to do approximately  $10^5$  multiplications per seconds and thus the computational complexity of both Alice and Bob is tolerable. A similar analysis applies for Protocol 4. Here, Alice and Bob must pre-compute  $N$  encryptions. Hence, we can conclude that the computational complexity is not a serious downside of the proposed protocols. Similar, although not as efficient, optimisation tricks can also be used to speed up Protocol 3 when  $\mathbf{x}$  and  $\mathbf{y}$  are not binary.

**Estimated communication complexity.** The only serious drawback of the new protocols is the communication overhead: since Alice sends  $N$  ciphertexts  $c_i$ , the overhead is  $k'/\mu$ , where  $k'$  is just the size of each ciphertext in bits. When using any of the currently known most efficient semantically secure homomorphic cryptosystems (e.g., the one from [Pai99]),  $k' \approx 2048$ . For  $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_{m'}$  with very small  $m'$ —say,  $m' \leq 13$ , this compares non-favourably with the overhead of the (insecure) Du-Atallah protocol which has the overhead of approximately 160 times with  $d = 80$  and  $k = 1024$ . For a large  $m'$ , the described protocol is already more communication-efficient than the Du-Atallah protocol.

**Comparison with Freedman-Nissim-Pinkas protocol.** Recently, Freedman, Nissim and Pinkas proposed a related cryptographically secure protocol for computing the set intersection cardinality [FNP04], a task that is equivalent to privately computing the scalar product of two binary vectors. In the non-shared case, the Freedman-Nissim-Pinkas protocol is more efficient than the new one, but then the participants also learn the values  $\text{supp}(\mathbf{x})$  and  $\text{supp}(\mathbf{y})$ . However, recall that in the data mining applications it is preferable that both parties will get only shares  $s_A + s_B = \mathbf{x} \cdot \mathbf{y} \pmod m$  of the scalar product, otherwise frequency of some infrequent sets is revealed. Moreover, sometimes only a list of frequent sets without frequencies is required.

Freedman, Nissim and Pinkas proposed also a solution for shared version, but their protocol requires a secure circuit evaluation. Briefly, secure evaluation means that first Alice and Bob obtain  $\text{supp}(\mathbf{x})$  different shares

$$s_i + t_i = \begin{cases} 0, & \text{if } x_i = 1, y_i = 1 \\ r_i, & \text{if } x_i = 1, y_i = 0 \end{cases} \pmod m$$

where  $r_i \in \mathbb{Z}_m$  is a random value and  $m$  is (say) a 1024-bit number. To securely compute  $\mathbf{x} \cdot \mathbf{y}$  by secure circuit evaluation, one therefore needs to execute oblivious transfer for each  $1024 \cdot \text{supp}(\mathbf{x})$  input bit pairs  $(s_i, t_i)$ . Since a  $\binom{2}{1}$ -oblivious transfer protocol requires sending at least three encryptions, the communication overhead of the Freedman-Nissim-Pinkas protocol is lower than the communication overhead of Protocol 3 only if  $\text{supp}(\mathbf{x}) \leq N/(3 \cdot 1024)$ , i.e., if the candidate set is very infrequent.

**Reducing communication overhead.** We shall now discuss how to reduce the overhead if it is known that  $\mathbf{x}$  and  $\mathbf{y}$  are Boolean. Again, similar optimisation techniques can be used when  $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_{\mu'}$  for some  $2 < \mu' \ll \mu$ . In the

following we assume that the plaintext space of the cryptosystem  $\Pi$  is a residue ring  $\mathbb{Z}_m$  such that  $\log m \geq 1024$ . This is the case for all widely known homomorphic cryptosystems. When we assume that  $x_i, y_i \in \{0, 1\}$ , every ciphertext  $c_i$  in Protocol 3 only transfers a single bit  $x_i$ , which results in communication overhead.

The next technique for packing several bits into one plaintext is fairly standard in cryptography (it has been used at least in the context of electronic voting [CGS97,DJ01], electronic auctions [LAN02] and oblivious transfer [Lip04]). To pack  $k$  entries into a single message—recall that the plaintext length is  $k$  bits—, we fix a radix  $B > N$ , such that  $B^k < m$ , and work implicitly with  $B$ -ary numbers. Let  $[v_k, \dots, v_2, v_1] = v_1 + v_2B + \dots + v_kB^{k-1}$ . Our method works only in the case when Alice and Bob do batch computation of scalar products, more precisely, when Alice and Bob need to compute  $\mathbf{x}_i \cdot \mathbf{y}$  for several vectors  $\mathbf{x}_i, i \in \{1, \dots, k\}$ , owned by Alice. (This is exactly what happens in the context of frequent itemset mining.)

The new batch scalar product protocol looks exactly like Protocol 3, except that Alice computes  $c_i$  as

$$\begin{aligned} c_i &= \text{Enc}_{\text{pk}}([x_{ki}, \dots, x_{2i}, x_{1i}]; r_i) \\ &= \text{Enc}_{\text{pk}}(1; 0)^{x_{1i}} \text{Enc}_{\text{pk}}(B; 0)^{x_{2i}} \dots \text{Enc}_{\text{pk}}(B^{k-1}; 0)^{x_{ik}} \text{Enc}_{\text{pk}}(0; r_i) . \end{aligned}$$

It takes at most  $k$  multiplications to compute  $c_i$ . Again, the encryptions  $\text{Enc}_{\text{pk}}(B^j; 0)$  can be computed in the setup phase. Hence, during the first step, Alice's computation is  $N$  encryptions and  $O(kN)$  multiplications.

At the second step of the protocol, Bob computes

$$\begin{aligned} w &= \prod_{i=1}^N \text{Enc}_{\text{pk}}(y_i[x_{ki}, \dots, x_{2i}, x_{1i}]; r_i) \text{Enc}_{\text{pk}}(-s_B; r') \\ &= \text{Enc}_{\text{pk}}([\mathbf{x}_k \cdot \mathbf{y}, \dots, \mathbf{x}_1 \cdot \mathbf{y}] - s_B; r'') . \end{aligned}$$

Hence, if Bob reveals  $s_B$ , Alice can restore *all* scalar products  $\mathbf{x}_j \cdot \mathbf{y}$ . Sometimes it is also needed that Alice be able only to compute  $\mathbf{x}_j \cdot \mathbf{y}$  for  $j \in I$ , where  $I$  is a proper subset of  $\{1, \dots, k\}$ . One can do this efficiently by using standard cryptographic techniques.

Therefore, when using the Paillier cryptosystem, the resulting protocol for privately computing the scalar product of two binary vectors has almost optimal communication overhead of  $\lceil \log N \rceil$  times. (When using the Damgård-Jurik cryptosystem, the communication overhead might even be smaller.) This should be compared to the 160 times overhead of the insecure Du-Atallah protocol.

**Related work.** After the preproceedings version of the current paper was published, we were acknowledged by Rebecca Wright of some previous work. In particular, in [WY04], Wright and Yang proposed essentially the same SSP protocol as Protocol 3, optimised for the case of binary data as in our paper. However, they did not consider the batch SSP case. (See [WY04] and the references therein.)

**Security in malicious model.** Protocol 3 can be made secure in the malicious model by letting Alice to prove in zero-knowledge, for every  $i$ , that  $c_i$  encrypts a value from  $\mathbb{Z}_\mu$ . This can be done efficiently in the random oracle (or common reference string) model [Lip03]. An alternative is to use conditional disclosure of secrets [AIR01] modified recently to the setting of Paillier’s cryptosystem in [LL05]. Both methods guarantee that at the end of a protocol run, Alice is no better off than mounting the next *probing attack*: Alice creates a suitable valid input vector  $\mathbf{x}'$ , executes the protocol with Bob, and obtains  $\mathbf{x}' \cdot \mathbf{y}$ . If  $\mathbf{x}'$  is suitably chosen (e.g.,  $x'_i = 1$  and  $x'_j = 0$  for  $j \neq i$ ), this may result in privacy leakage. However, this probing attack is unavoidable, no matter what private scalar product protocol is used instead of Protocol 3. The only way to tackle this attack is to let Alice to prove that her input  $\mathbf{x}$  is “correctly” computed, whatever “correctly” means in the concrete application (e.g., in frequent itemset mining on vertically distributed databases). While such a functionality can be added to Protocol 3, it is not a part of the definition of a “private scalar product” protocol, but highly application-dependent (and thus should be left to be specified on a higher level), and very often, highly costly.

## 5 Conclusions

The secure computation of a scalar product is an important task within many data mining algorithms that require the preservation of privacy. Recently, several protocols have been proposed to solve this task. We have shown, however, that they are insecure. Moreover, we presented a private scalar product protocol based on standard cryptographic techniques and proved that it is secure. Furthermore, we described several optimisations in order to make it very efficient in practice.

**Acknowledgements.** We would like to thank Benny Pinkas for useful comments. The second and the third author were partially supported by the Estonian Information Technology Foundation, the Finnish Defence Forces Institute for Technological Research and by the Finnish Academy of Sciences.

## References

- [AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In Birgit Pfitzmann, editor, *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 119–135, Innsbruck, Austria, 6–10 May 2001. Springer-Verlag.
- [AMS<sup>+</sup>96] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast Discovery of Association Rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI/MIT Press, 1996.
- [BB00] Jean-Francois Boulicaut and Artur Bykowski. Frequent Closures as a Concise Representation for Binary Data Mining. In *PAKDD 2000*, volume 1805 of *Lecture Notes in Computer Science*, pages 62–73. Springer, 2000.

- [BBR03] Jean-Francois Boulicaut, Artur Bykowski, and Christophe Rigotti. Free-Sets: A Condensed Representation of Boolean Data for the Approximation of Frequency Queries. *Data Mining and Knowledge Discovery*, 7(1):5–22, 2003.
- [CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In Walter Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 103–118, Konstanz, Germany, 11–15 May 1997. Springer-Verlag.
- [DA01a] Wenliang Du and Mikhail J. Atallah. Privacy-Preserving Statistical Analysis. In *Proceedings of the 17th Annual Computer Security Applications Conference*, pages 102–110, New Orleans, Louisiana, USA, December 10–14 2001.
- [DA01b] Wenliang Du and Mikhail J. Atallah. *Protocols for Secure Remote Database Access with Approximate Matching*, volume 2 of *Advances in Information Security*, page 192. Kluwer Academic Publishers, Boston, 2001. <http://www.wkap.nl/prod/b/0-7923-7399-5>.
- [DJ01] Ivan Damgård and Mads Jurik. A Generalisation, a Simplification and Some Applications of Paillier’s Probabilistic Public-Key System. In Kwangjo Kim, editor, *Public Key Cryptography 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136, Cheju Island, Korea, 13–15 February 2001. Springer-Verlag.
- [DZ02] Wenliang Du and Zhijun Zhan. A Practical Approach to Solve Secure Multi-party Computation Problems. In Carla Marceau and Simon Foley, editors, *Proceedings of New Security Paradigms Workshop*, pages 127–135, Virginia Beach, Virginia, USA, September 23–26 2002. ACM Press.
- [FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 1–19, Interlaken, Switzerland, 2–6 May 2004. Springer-Verlag.
- [Gol04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.
- [LAN02] Helger Lipmaa, N. Asokan, and Valtteri Niemi. Secure Vickrey Auctions without Threshold Trust. In Matt Blaze, editor, *Financial Cryptography — Sixth International Conference*, volume 2357 of *Lecture Notes in Computer Science*, pages 87–101, Southampton Beach, Bermuda, 11–14 March 2002. Springer-Verlag.
- [Lip03] Helger Lipmaa. On Diophantine Complexity and Statistical Zero-Knowledge Arguments. In Chi Sung Lai, editor, *Advances on Cryptology — ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, pages 398–415, Taipei, Taiwan, 30 November–4 December 2003. Springer-Verlag.
- [Lip04] Helger Lipmaa. An Oblivious Transfer Protocol with Log-Squared Total Communication. Technical Report 2004/063, International Association for Cryptologic Research, February 25 2004.
- [LL04] Sven Laur and Helger Lipmaa. On Private Similarity Search Protocols. In Sanna Liimatainen and Teemupekka Virtanen, editors, *Proceedings of the Ninth Nordic Workshop on Secure IT Systems (NordSec 2004)*, pages 73–77, Espoo, Finland, November 4–5, 2004.
- [LL05] Sven Laur and Helger Lipmaa. Additive Conditional Disclosure of Secrets. Manuscript, January 2005.

- [Pai99] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, 2–6 May 1999. Springer-Verlag.
- [PHM00] J. Pei, J. Han, and R. Mao. CLOSET: An efficient algorithm for mining frequent closed itemsets. In *2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2000.
- [Pin02] Benny Pinkas. Cryptographic Techniques for Privacy-Preserving Data Mining. *KDD Explorations*, 4(2):12–19, 2002.
- [VC02] Jaideep Vaidya and Chris Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. In *Proceedings of The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 639–644, Edmonton, Alberta, Canada, July 23–26 2002. ACM.
- [vLW92] Jacobus H. van Lint and Richard M. Wilson. *A Course in Combinatorics*. Cambridge University Press, 1992.
- [WY04] Rebecca N. Wright and Zhiqiang Yang. Privacy-Preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data. In *Proceedings of The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 713–718, Seattle, Washington, USA, August 22–25 2004. ACM.

## A Private Generalised Scalar Product Protocol

Next, we propose a secure generalised scalar product protocol (Protocol 4) for

$$\langle \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k \rangle = \sum_{i=1}^N x_{1i} \cdots x_{ki} .$$

For the sake of simplicity, we consider only the three-party case but the protocol can be easily generalised. Again, Alice has a private key; Bob and Carol know only the corresponding public key. The security of the generalised scalar product protocol depends on Alice. Namely, when Alice colludes with other parties then privacy can be compromised. For example, colluding Alice and Carol can reveal  $y_i$ , unless  $x_i = 0$ , since  $\text{Dec}_{\text{sk}}(d_i) = x_i y_i$ . Thus, we get the following result.

**Theorem 2.** *Assume that  $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$  is a semantically secure homomorphic public-key cryptosystem with  $P(\text{sk}) = \mathbb{Z}_m$  for some large  $m$ . Protocol 4 is a secure generalised scalar product protocol. In particular, it is secure against all possible coalitions provided that Alice does not collude with other parties.*

The proof is a simple generalisation of the previous proof. Bob must randomise  $c_i$ 's as  $d_i = c_i \cdot \text{Enc}_{\text{pk}}(0; r'_i)$ , since otherwise the values of  $y_i$ 's can be detected only by comparing the ciphertext that he receives from Alice with the one he sends to Carol. The sharing step 4 allows combine the outcome with other cryptographic protocols.

The assumption that Alice does not collude with other parties is quite strong. When we modify the protocol so that  $(\text{sk}, \text{pk})$  is generated jointly by Alice, Bob and Carol and that on the step 4, they do threshold decryption of  $w$ , we get a private SP protocol with the next security result:

PRIVATE INPUTS: Private vectors  $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{Z}_\mu^N$ .  
PRIVATE OUTPUTS: Shares  $s_A + s_B + s_C \equiv \langle \mathbf{x}, \mathbf{y}, \mathbf{z} \rangle \pmod{m}$

1. Alice does:  
Generate a key-pair  $(\mathbf{sk}, \mathbf{pk})$ .  
Send the public key  $\mathbf{pk}$  to Bob and Carol.
2. Alice does for  $i \in \{1, \dots, N\}$ :  
Send  $c_i = \mathbf{Enc}_{\mathbf{pk}}(x_i; r_i)$  to Bob.
3. Bob does for  $i \in \{1, \dots, N\}$ :  
Set  $d_i = c_i^{y_i} \mathbf{Enc}_{\mathbf{pk}}(0; r'_i)$ .  
Send  $d_i$  to Carol.
4. Carol does:  
Set  $w \leftarrow \prod_{i=1}^N c_i^{z_i}$ .  
Generate a random plaintext  $s_C$  and a random nonce  $r'$ .  
Send  $w' \leftarrow w \cdot \mathbf{Enc}_{\mathbf{pk}}(-s_C; r')$  to Bob.
5. Bob does:  
Generate a random plaintext  $s_B$  and a random nonce  $r''$ .  
Send  $w'' \leftarrow w' \cdot \mathbf{Enc}_{\mathbf{pk}}(-s_B; r'')$  to Alice.
6. Alice computes  $s_A \leftarrow \mathbf{Dec}_{\mathbf{sk}}(w'') = \mathbf{x} \cdot \mathbf{y} - s_B - s_C$ .

**Protocol 4:** Private generalised homomorphic SSP protocol

**Theorem 3.** *Assume  $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$  is a semantically secure homomorphic threshold public-key cryptosystem. Then Protocol 4, generalised to  $\kappa$  parties, is secure against coalitions by  $< \kappa/2$  parties.*



## **Publication P2**

Sven Laur and Helger Lipmaa. A New Protocol for Conditional Disclosure of Secrets and Its Applications. Lecture Notes in Computer Science, volume 4521, pages 207–225. © Springer-Verlag. Reprinted with permission.



# A New Protocol for Conditional Disclosure of Secrets And Its Applications

Sven Laur<sup>1</sup> and Helger Lipmaa<sup>2</sup>

<sup>1</sup> Helsinki University of Technology, Finland

<sup>2</sup> University College London, UK

**Abstract.** Many protocols that are based on homomorphic encryption are private only if a client submits inputs from a limited range  $\mathcal{S}$ . Conditional disclosure of secrets (CDS) helps to overcome this restriction. In a CDS protocol for a set  $\mathcal{S}$ , the client obtains server's secret if and only if the client's inputs belong to  $\mathcal{S}$  and thus the server can guard itself against malformed queries. We extend the existing CDS protocols to work over additively homomorphic cryptosystems for every set from  $\mathbf{NP}/\mathbf{poly}$ . The new construction is modular and easy to apply. As an example, we derive a new oblivious transfer protocol with log-squared communication and a millionaire's protocol with logarithmic communication. We also implement private, universally verifiable and robust multi-candidate electronic voting so that all voters only transmit an encryption of their vote. The only hardness assumption in all these protocols is that the underlying public-key cryptosystem is IND-CPA secure and the plaintext order does not have small factors.

**Keywords.** Conditional disclosure of secrets, crypto-computing, homomorphic encryption, oblivious transfer, two-party computation.

## 1 Introduction

Homomorphic encryption is a powerful tool that provides efficient private implementations for many basic operations such as scalar product, oblivious transfer and oblivious polynomial evaluation. However, basic versions of these protocols without zero-knowledge proofs of correctness are secure only in a semihonest model, where all parties submit inputs from a limited range, and are not protected against malicious behaviour. Consequently, a malicious adversary can completely or partially learn the secret inputs. Conditional disclosure of secrets [GIKM00,AIR01], also known as input verification gadget [BGN05], is a protection mechanism against such attacks. Unfortunately, current solutions [AIR01,BGN05] are secure only if the plaintext space has a prime order, whereas most additively homomorphic encryption schemes have a composite plaintext order. We provide the first conditional disclosure of secrets protocol that works in conjunction with *all* currently known additively homomorphic encryption schemes. Hence, we can efficiently and more securely solve many practical problems.

Formally, we consider only two-party protocols between a client and a server, though our results can be extended to the multiparty setting. At the end of such a protocol the client should learn the desired value whereas the server should learn nothing. Our main goal is to achieve *relaxed-security*; that is, the protocol must be secure against malicious clients and semihonest servers. Such a model is widely used in current cryptographic

literature [NP99,AIR01] and is well-justified in practical applications: as the number of possible service providers is relatively small compared to the clients, it is possible to force semihonest behaviour with auditing. Moreover, service providers must preserve their reputation and thus they are less likely to act maliciously.

For clarity and brevity, we state our main results in the public key model, where the client is guaranteed to know a valid secret key and the server knows the corresponding public key. The choice of the model is not too restrictive: with a proper initialisation phase all our protocols can be implemented in the standard model, see Sect. 7. On the other hand, such a model enables to prove security of parallel compositions. Composability together with our new basic construction leads to a simpler and more modular way to construct complex protocols. Shortly put, relaxed-security follows directly from the protocol design and there is no need to handcraft the proof. More precisely, we show how to decompose a protocol into elementary tasks that can be efficiently implemented with any additively homomorphic IND-CPA secure cryptosystem, provided that the plaintext order does not have unknown small factors.

In Sect. 3, we establish basic security notions and derive a necessary machinery to analyse parallel compositions. The core results of our papers are presented in Sect. 4. We note that most existing additively homomorphic protocols are based on the possibility of computing the next three basic primitives on ciphertexts: addition of ciphertexts, multiplication with a constant, and *disclose-if-equal* (DIE). In a disclose-if-equal protocol, the server obviously releases secret  $\beta$  only if the client sends a valid encryption of  $x$ , where the coefficient  $x$  can be freely chosen by the server. The current cryptographic literature is full of many useful and efficient two-message protocols that are based on these three primitives. Unfortunately, the standard DIE protocol defined say in [AIR01], and then used in many subsequent papers, is secure only if the plaintext space has a prime order and thus can only be used in conjunction with the lifted ElGamal cryptosystem where one has to compute discrete logarithms to decrypt. We provide a new DIE protocol that works in conjunction with *all* currently known additively homomorphic encryption schemes. As a result, we can naturally simplify or extend many protocols that utilise the DIE functionality, e.g. [AIR01,Ste98,Lip05,BK04,FNP04,LLM05].

The rest of the paper provides many useful applications of these generic building blocks. In Sect. 5, we present a two-message protocol for conditional disclosure of secrets (CDS), where the client learns a secret  $\beta$  only if his message  $q$  is a valid encryption of  $x \in \mathcal{S}$ , where  $\mathcal{S}$  is a publicly known set. Hence, the server can use  $\beta$  as a one-time pad to protect the protocol output, i.e., the client learns nothing unless  $\text{Dec}_{\text{sk}}(q) \in \mathcal{S}$ . The latter forms a basis of the CDS transformation that can guard any two-message protocol, where the first message is a vector of ciphertexts, against malicious clients. A slightly extended CDS construction provides an efficient solution to the millionaire problem and conditional oblivious transfer. Another extension of CDS provides a way to implement electronic voting and auctions without non-interactive zero-knowledge proofs in the multi-party setting using threshold-decryption. Finally, we compare our results with conventional cryptographic methods to provide some interesting insights and show the theoretical significance of our results, see Sect. 7.

**History.** The new DIE protocol, together with the CDS protocol and the CDS transformation date from August 2004 and has been available on eprint since 2005.

## 2 Cryptographic Preliminaries

**Distributions.** For a finite set  $X$ , let  $\mathcal{U}(X)$  denote the uniform distribution over  $X$  and  $x \leftarrow X$  denote a uniform draw from  $X$ . Two distributions  $\mathcal{D}_1$  and  $\mathcal{D}_2$  over a discrete support  $X$  are statistically  $\varepsilon$ -close,  $\mathcal{D}_1 \stackrel{\varepsilon}{\sim} \mathcal{D}_2$ , if their statistical difference  $\max_{S \subseteq X} |\Pr[\mathcal{D}_1 \in S] - \Pr[\mathcal{D}_2 \in S]| \leq \varepsilon$ . A shorthand  $\mathcal{D}_1 \equiv \mathcal{D}_2$  denotes  $\mathcal{D}_1 \stackrel{0}{\sim} \mathcal{D}_2$ .

**Homomorphic encryption.** A public-key cryptosystem  $\pi$  is defined by three algorithms. A key generation algorithm  $\text{Gen}$  returns a secret and public key pair  $(\text{sk}, \text{pk})$ . Corresponding  $\text{Enc}_{\text{pk}}(\cdot)$  and  $\text{Dec}_{\text{sk}}(\cdot)$  algorithms are used to encrypt and decrypt messages. Let  $\mathcal{M}$  and  $\mathcal{C}$  denote the corresponding message and ciphertext spaces. Then we require  $\text{Dec}_{\text{sk}}(\text{Enc}_{\text{pk}}(x)) = x$  for every  $x \in \mathcal{M}$  and assume that there exists efficient membership test for the ciphertext space  $\mathcal{C}$ . Privacy of encrypted messages is guaranteed by IND-CPA security. For any *stateful* probabilistic algorithm  $A$ , its IND-CPA advantage quantifies the ability to distinguish ciphertexts:

$$\text{Adv}_{\pi}^{\text{IND-CPA}}(A) = 2 \cdot \left| \Pr \left[ \begin{array}{l} (\text{sk}, \text{pk}) \leftarrow \text{Gen}, (x_0, x_1) \leftarrow A(\text{pk}), i \leftarrow \{0, 1\} \\ c \leftarrow \text{Enc}_{\text{pk}}(x_i) : A(x_0, x_1, c) = i \end{array} \right] - \frac{1}{2} \right|,$$

where the probability is taken over coin tosses of all relevant algorithms. A cryptosystem  $\pi$  is  $(\varepsilon, \tau)$ -IND-CPA-secure if  $\text{Adv}_{\pi}^{\text{IND-CPA}}(A) \leq \varepsilon$  for any  $\tau$ -time adversary  $A$ .

A cryptosystem  $\pi$  is *additively homomorphic*, if  $\mathcal{M} = \mathbb{Z}_N$  for some  $N$ , and for any  $(\text{sk}, \text{pk}) \leftarrow \text{Gen}$  and valid messages  $x_1, x_2 \in \mathcal{M}$  the distribution of products  $\text{Enc}_{\text{pk}}(x_1) \cdot \text{Enc}_{\text{pk}}(x_2)$  coincides with the distribution of ciphertexts  $\text{Enc}_{\text{pk}}(x_1 + x_2)$ . To be precise, the equivalence

$$\text{Enc}_{\text{pk}}(x_1) \cdot \text{Enc}_{\text{pk}}(x_2) \equiv \text{Enc}_{\text{pk}}(x_1 + x_2)$$

must hold for any fixed ciphertext  $\text{Enc}_{\text{pk}}(x_1)$ . That is, given  $\text{Enc}_{\text{pk}}(x_1) \cdot \text{Enc}_{\text{pk}}(x_2)$ , even an unbounded adversary learns nothing beyond  $x_1 + x_2$ . A cryptosystem  $\pi$  is *multiplicatively homomorphic*, if  $\text{Enc}_{\text{pk}}(x_1) \cdot \text{Enc}_{\text{pk}}(x_2) \equiv \text{Enc}_{\text{pk}}(x_1 \cdot x_2)$  for any  $(\text{sk}, \text{pk}) \leftarrow \text{Gen}$  and  $x_1, x_2 \in \mathcal{M}$ , where  $\mathcal{M}$  is a multiplicative group where computing the discrete logarithm is hard. In many practical applications, multiplicatively homomorphic cryptosystems  $\text{Enc}$  are converted to additively homomorphic cryptosystems  $\overline{\text{Enc}}$  by using the lifted encryption rule  $\overline{\text{Enc}}_{\text{pk}}(x) := \text{Enc}_{\text{pk}}(g^x)$ . Such lifted cryptosystems have reduced utility, as the new decryption rule requires computation of discrete logarithms and one can successfully decrypt only a small fraction of ciphertexts.

Many well-known homomorphic cryptosystems are IND-CPA secure under reasonable complexity assumptions, e.g. [Elg85, Pai99, DJ01]. Existing additively homomorphic cryptosystems have a composite plaintext order with large factors. For example, the plaintext order of the Paillier cryptosystem [Pai99] is an RSA modulus and thus its smallest prime factor is approximately  $\sqrt{N}$ . The Goldwasser-Micali cryptosystem [GM82] is the only known exception, as it is additively homomorphic over  $\mathbb{Z}_2$ . Such plaintext space is too small for many applications. All known cryptosystems with a large prime plaintext order are multiplicative, e.g., the ElGamal cryptosystem [Elg85].

### 3 Basic Properties of Two-Message Protocols

Throughout the paper, we consider two-message protocols where a client sends a query  $q$  to a server that replies with  $a$ , and then the client computes a desired output from  $a$ . The server should learn nothing about the query. The client should learn  $f(\alpha, \beta)$ , where  $\alpha$  denotes client's private input vector and  $\beta$  denotes server's private input vector. Mostly, we consider the relaxed-security against unbounded clients and computationally bounded servers, but sometimes we consider also the setting where both parties are computationally bounded. A protocol is *correct* if the client always recovers  $f(\alpha, \beta)$  when both parties are honest. A priori we do not assume correctness from all protocols, as sometimes it is sufficient to know that a client cannot learn anything beyond  $f(\alpha, \beta)$ .

In the simplest case, the query  $q$  consists of encrypted inputs  $(\alpha_1, \dots, \alpha_m)$  and the server uses properties of additively homomorphic encryption to compose an appropriate reply. We call such protocols *additively homomorphic two-message protocols*. Here, we explicitly assume that the server knows public key  $pk$  and thus can efficiently verify that the query consists of valid ciphertexts and ignore malformed queries. Notably, many interesting tasks can be solved with additively homomorphic two-message protocols. Computationally-private information retrieval [AIR01,Ste98,Lip05], solutions to millionaire's problem [BK04,Fis01], and various protocols for privacy-preserving data mining tasks [FNP04,WY04,GLLM04] form only a small set of such protocols.

**Relaxed-security in the PKI model.** As usual, we define security by comparing the real and ideal model. However, we explicitly assume that the client knows the secret key, the server knows the corresponding public key and only the client can deviate from the protocol specification. Formally, a trusted key generator initially runs the key generation algorithm  $\text{Gen}$  for a cryptosystem  $\pi$ , and then privately sends  $(sk, pk)$  to the client and  $pk$  to the server. In particular, the server knows that  $pk$  corresponds to this fixed client. This key pair is then possibly used in many different protocol runs.

Note that the PKI model is normal and even desirable in many applications, e.g. e-voting. Still, we stress that we use the PKI model only for the sake of simplicity of security proofs. In Sect. 7, we show how to replace the trusted key generator by a key transfer protocol with a marginal degradation of security.

Since the server obtains no output and is always semihonest, we can decompose the standard security definition into two orthogonal requirements: client-privacy and server-privacy. A two-message protocol is  $(\varepsilon, \tau)$ -*client-private*, if for any  $\tau$ -time stateful adversary  $A$ , the next inequality holds:

$$2 \cdot \left| \Pr \left[ \begin{array}{l} (sk, pk) \leftarrow \text{Gen}, (\alpha_0, \alpha_1) \leftarrow A(pk), \\ i \leftarrow \{0, 1\}, q \leftarrow q_{pk}(\alpha_i) : A(\alpha_0, \alpha_1, q) = i \end{array} \right] - \frac{1}{2} \right| \leq \varepsilon ,$$

where  $q_{pk}(\alpha_i)$  denotes the first message computed by the honest client. Server-privacy has a slightly more complicated definition, since we must transform any efficient adversary from the real world to an efficient adversary in the ideal model, where a trusted third party (TTP) computes  $f(\alpha, \beta)$ . Hence, the definition incorporates a simulator  $\text{Sim}$  and a distinguisher  $B$  and we need to explicitly quantify their efficiency. The simulator

Sim gets  $(\text{sk}, q)$  as an input and can send  $\alpha^*$  once to the TTP. Then Sim obtains the value of  $f^* = f(\alpha^*, \beta)$  and can proceed with the simulation. For brevity, let us define

$$p_r = \Pr [(\text{sk}, \text{pk}) \leftarrow \text{Gen}, (\beta, q) \leftarrow A(\text{sk}), \alpha \leftarrow \alpha_{\text{pk}}(q, \beta) : B(\beta, q, \alpha) = 1] ,$$

$$p_i = \Pr [(\text{sk}, \text{pk}) \leftarrow \text{Gen}, (\beta, q) \leftarrow A(\text{sk}), \hat{\alpha} \leftarrow \text{Sim}_{\text{sk}}(q, f^*) : B(\beta, q, \hat{\alpha}) = 1] ,$$

where  $\alpha(q, \beta)$  denotes the answer of the honest server with the input  $\beta$  to the query  $q$ . A protocol implements  $(\tau, \delta, t, \varepsilon)$ -server-privately a function  $f$ , if for any  $\tau$ -time adversary  $A$  there exists a  $(t + \delta)$ -time simulator Sim such that  $|p_r - p_i| \leq \varepsilon$  for any  $t$ -time distinguisher  $B$ . In the information-theoretical setting, algorithms  $A$ , Sim and  $B$  are unbounded. A protocol is  $\varepsilon$ -server-private if for any adversary  $A$  there exists a simulator Sim such that their output distributions are statistically  $\varepsilon$ -close. We say that a protocol is  $(\varepsilon_1, \tau; \varepsilon_2)$ -relaxed-secure if it is  $(\varepsilon_1, \tau)$ -client-private and  $\varepsilon_2$ -server-private. Relaxed-security is widely used standard security assumption, see [NP99,AIR01].

**Extractability and simulatability.** Usually, the client-privacy follows directly from security assumptions. For example, additively homomorphic protocols are client-private by the construction, provided that the cryptosystem is IND-CPA secure. Proofs of server-privacy can be significantly simplified by considering the following notions of extractability and simulatability. As client can be malicious, the simulator Sim must somehow deduce the intended input  $\alpha^*$ . In the PKI model, the simulator can use  $\text{sk}$  to determine the input  $\alpha^*$  directly from  $q$ . A two-message protocol is *extractable* if there exists an efficient algorithm  $\text{Ext}_{\text{sk}}(\cdot)$  such that  $\text{Ext}_{\text{sk}}(q_{\text{pk}}(\alpha)) = \alpha$  for all valid inputs and  $\text{Ext}_{\text{sk}}(q) = \perp$  for all invalid queries  $q$  that do not correspond to any input.

In many protocols, the server's reply can be perfectly or almost perfectly simulated knowing only the corresponding client's output  $f^*$  and a secret key  $\text{sk}$ . We formalise this as simulatability. Consider a protocol transcript  $(q, \alpha)$  between the honest client and server. Let  $f^* = f(\alpha, \beta)$  be the corresponding client's output. Then the server's reply is  $\varepsilon_2$ -simulatable if there exists an efficient algorithm  $\text{Sim}_{\text{sk}}^*(\cdot)$  such that the output distributions  $(q, \alpha)$  and  $(q, \hat{\alpha})$  are statistically  $\varepsilon_2$ -close even for a fixed  $q$ , where  $\hat{\alpha} \leftarrow \text{Sim}_{\text{sk}}^*(q, f^*)$ . The notion of  $(t, \varepsilon_2)$ -simulatability is defined analogously. Extractability together with simulatability implies server-privacy:

**Theorem 1.** *If a two-message protocol is extractable,  $\varepsilon_2$ -simulatable and the server ignores malformed queries, then the protocol is also  $\varepsilon_2$ -server-private in the PKI model.*

*Proof.* We construct a universal simulator Sim as follows. If the query  $q$  is malformed then the simulator ignores it. Otherwise, Sim extracts the intended input  $\alpha^* \leftarrow \text{Ext}_{\text{sk}}(q)$  and sends  $\alpha^*$  to the TTP. Given the reply  $f^* = f(\alpha^*, \beta)$  from the TTP, the simulator uses  $\text{Sim}_{\text{sk}}^*(q, f^*)$  to simulate the reply  $\hat{\alpha}$ . Since malformed queries are discarded in both worlds, the distributions  $(\beta, q, \alpha)$  and  $(\beta, q, \hat{\alpha})$  are statistically  $\varepsilon_2$ -close.  $\square$

**Forked composition.** We can use Thm. 1 to prove that a parallel composition of extractable and simulatable protocols preserves server-privacy. It makes sense to consider protocols that share the query phase as we can always merge different queries into a single query. Let two-message protocols  $\Pi_1, \dots, \Pi_s$  share the first message  $q$ . Then the *forked composition*  $\text{Forked}[\Pi_1, \dots, \Pi_s]$  is defined as follows:

1. The client computes the query  $q$  and sends it to the server.
2. The server uses  $q$  to compute replies  $a_1, \dots, a_s$  according to  $\Pi_1, \dots, \Pi_s$ .
3. The server sends  $a_1, \dots, a_s$  to the client.
4. The client computes the private output  $(f_1, \dots, f_s)$  according to  $\Pi_1, \dots, \Pi_s$ .

It is easy to prove that a client can learn nothing beyond  $f_1(\alpha, \beta), \dots, f_s(\alpha, \beta)$ .

**Theorem 2.** *Let  $\Pi_1, \dots, \Pi_s$  be extractable and respectively  $\varepsilon_i$ -simulatable implementations of functionalities  $f_i$ . Then the composition  $\text{Forked}[\Pi_1, \dots, \Pi_s]$  is an extractable and  $(\varepsilon_1 + \dots + \varepsilon_s)$ -simulatable implementation of the functionality  $f = (f_1, \dots, f_s)$ .*

*Proof.* Extractability is clear. By the definition of simulatability, there exist simulators  $\text{Sim}_{\text{sk},i}^*$  that output simulated replies  $\hat{a}_i$  such that  $(q, a_i)$  and  $(q, \hat{a}_i)$  are statistically  $\varepsilon_i$ -close even for fixed  $q$ . Now, define a simulator  $\text{Sim}_{\text{sk}}^*$  that given  $q$  and  $f^* = (f_1(\alpha^*, \beta), \dots, f_s(\alpha^*, \beta))$  runs  $\text{Sim}_{\text{sk},i}^*(q, f_i^*)$  for  $i \in \{1, \dots, s\}$  and outputs  $\hat{a}_1, \dots, \hat{a}_s$ . By the construction, the distributions  $(q, a_1, \dots, a_s)$  and  $(q, \hat{a}_1, \dots, \hat{a}_s)$  are statistically  $(\varepsilon_1 + \dots + \varepsilon_s)$ -close even for a fixed  $q$  and the simulatability follows.  $\square$

**Reducing communication further with CPIR.** In many two-message protocols, the client must access only a short part of the reply  $a$  to recover the output  $f(\alpha, \beta)$  whereas the rest of  $a$  consists of random noise. Hence, we can significantly decrease the total communication  $|q| + |a|$ , if the client could fetch only useful parts of  $a$ . The latter can be done using *computationally private information retrieval* (CPIR). In a 1-out-of- $n$  CPIR protocol, the server maintains a database  $\beta = (\beta_1, \dots, \beta_n)$  of  $\ell$ -bit strings and the client can fetch  $\beta_i$  so that a computationally bounded server learns nothing. The basic properties of CPIR protocols are determined by parameters  $n$  and  $\ell$ . It is trivial to achieve communication complexity  $\Theta(n\ell)$  just by sending the whole database so one considers only CPIR protocols with sublinear communication. There is a wide range of such protocols. Recent protocols achieve communication that is low-degree polylogarithmic in the database size, see [Lip05,GR05] for further references.

Now, assume that the server's reply has a structure  $a = (a_1, \dots, a_n)$  and the client needs to recover at most  $t$  elements. Then the client can use  $t$  parallel CPIR queries to fetch desired parts  $a_{i_1}, \dots, a_{i_t}$ . Note that the CPIR queries can be sent together with the protocol  $\Pi$  messages, provided that the CPIR instance is run independently from  $\Pi$  or joining queries does not decrease client-privacy. Server-privacy cannot decrease, as the replies of CPIR queries are computed from the original reply  $a$ .

## 4 Three Basic Crypto-Computing Primitives

“Crypto-computing” is often used to describe two-message protocols, where a server uses some basic operations on client's garbled inputs to compute reply that reveals only  $f(\alpha, \beta)$ . The first comparative study [SYY99] showed how to crypto-compute predicates with logarithmic circuit depth using the Goldwasser-Micali cryptosystem. Later, this construction was somewhat generalised to compute the greater-than predicate [Fis01]. Here, we provide three basic crypto-computing primitives for additively homomorphic cryptosystems with large factors of the plaintext space. Note that the

server can crypto-compute ciphertexts of sums and products with one public factor obliviously from ciphertexts, as

$$\text{Enc}_{\text{pk}}(x_1 + x_2) \equiv \text{Enc}_{\text{pk}}(x_1) \cdot \text{Enc}_{\text{pk}}(x_2) \cdot \text{Enc}_{\text{pk}}(0) \quad (1)$$

$$\text{Enc}_{\text{pk}}(x \cdot y) \equiv \text{Enc}_{\text{pk}}(y)^x \cdot \text{Enc}_{\text{pk}}(0) \quad (2)$$

hold by the definition of additively homomorphic cryptosystems. Here the multiplication by  $\text{Enc}_{\text{pk}}(0)$  is necessary to re-randomise the replies.

But there is also a third generic operation that implicitly “tests” whether a ciphertext  $c$  is an encryption of  $x$ . The existence of this operation depends additionally on the order of the plaintext group. More precisely, a *disclose-if-equal* (DIE) protocol allows releasing of a secret  $\beta$  only if  $\text{Dec}_{\text{sk}}(c) = x$  where the server can freely choose  $x$ . The idealised functionality of DIE protocol is defined as follows

$$f(\alpha, \beta) = \begin{cases} \beta, & \text{if } \alpha = x \text{ ,} \\ \perp, & \text{if } \alpha \neq x \text{ .} \end{cases}$$

The simplest implementation of DIE protocol was given in the paper [AIR01]:

1. The client sends  $c \leftarrow \text{Enc}_{\text{pk}}(\alpha)$  to the server.
2. If  $c \in \mathcal{C}$  then the server sends a reply  $\mathbf{a} \leftarrow (c \cdot \text{Enc}_{\text{pk}}(-x))^r \cdot \text{Enc}_{\text{pk}}(\beta)$  for  $r \leftarrow \mathcal{M}$ .
3. The client outputs  $\text{Dec}_{\text{sk}}(\mathbf{a}) = (\alpha - x)r + \beta$ .

If the plaintext space has a prime order, then  $(\alpha - x)r$  has uniform distribution over  $\mathcal{M}$  when  $x \neq \alpha$ . Consequently, the protocol is perfectly simulatable: if  $f(\alpha, \beta) = \perp$  a simulator should output a random encryption  $\text{Enc}_{\text{pk}}(m)$  for  $m \leftarrow \mathcal{M}$  and  $\text{Enc}_{\text{pk}}(\beta)$  otherwise. Therefore, the basic DIE protocol is also relaxed-secure.

On the other hand, the protocol is not correct, since the client obtains a random output when  $\text{Dec}_{\text{sk}}(c) \neq x$ . If  $x$  is public then the correctness is not an issue, as the client knows whether  $\text{Dec}_{\text{sk}}(c) = x$  or not. Otherwise, the construction guarantees only that the client learns nothing about  $\beta$  when  $\text{Dec}_{\text{sk}}(c) \neq x$ . Moreover, if the server sets the first  $k$ -bits of  $\beta$  to 0, then the honest client can detect  $\alpha \neq x$  with failure probability  $2^{-k}$ , i.e., there is a trade-off between reliability and throughput.

Unfortunately, the basic DIE protocol is not secure if the message space has a composite order. As an example, consider the Paillier cryptosystem, where  $N = pq$  is an RSA modulus. If a malicious client sends  $c \leftarrow \text{Enc}_{\text{pk}}(p + x)$  then  $\text{Dec}_{\text{sk}}(\mathbf{a}) = \beta + rp \pmod N$  and the client can recover  $\beta \pmod p$  although  $\text{Dec}_{\text{sk}}(c) \neq x$ . Since the DIE protocol is a building block in many existing protocols, then such leakage might cause a domino effect that can completely reveal server’s input. For example, the circuit CDS protocol in Sect. 5 is extremely vulnerable against such attacks. Therefore, we devise a new DIE protocol that works in conjunction with all currently known additively homomorphic cryptosystems. As a result, we can naturally simplify many protocols [AIR01, Ste98, Lip05, BK04, FNP04, LLM05] that use the lifted ElGamal cryptosystem or zero-knowledge correctness proofs to guarantee security of the DIE protocol.

**New general construction for DIE.** Server-privacy of the basic DIE protocol hinges on the fact that  $\alpha\mathbb{Z}_N = \{\alpha r : r \in \mathbb{Z}_N\} = \mathbb{Z}_N$  for any  $\alpha \neq 0$ . If the message space

**Query phase:**

The client sends  $q \leftarrow \text{Enc}_{\text{pk}}(\alpha)$  to the server.

**Transfer phase:**

If the ciphertext is invalid  $q \notin \mathcal{C}$  then the server returns  $\perp$ .

Otherwise, the server returns  $a \leftarrow (c \cdot \text{Enc}_{\text{pk}}(-x))^r \cdot \text{Enc}_{\text{pk}}(\text{encode}(\beta))$  for  $r \leftarrow \mathcal{M}$ .

**Post-processing:**

The client computes  $y = \text{Dec}_{\text{sk}}(a)$  and returns  $\text{decode}(y)$ .

**Protocol 1:** Disclose-if-equal protocol of  $\ell$ -bit secrets for the constraint  $\text{Dec}_{\text{sk}}(q) = x$

contains non-trivial additive subgroups (ideals)  $\{0\} \neq \mathbb{G} \subseteq \mathbb{Z}_N$  then the client can choose a ciphertext  $c$  so that the reply  $a$  enables to restore the coset  $\beta + \mathbb{G}$ . Consequently, a malicious client can learn up to  $\log_2 N - \log_2 \Phi$  bits of information, where  $\Phi$  is the minimal size of the non-trivial subgroup  $\mathbb{G}$ . To seal the leakage, we must use a probabilistic encoding for  $\beta$  such that the total entropy of  $\mathbb{G}$  together with the encoding  $\text{encode}(\beta)$  is roughly  $\log_2 N$ . Let us define an encoding for  $\ell$ -bit strings

$$\begin{aligned} \text{encode}(\beta) &= \beta + 2^\ell \cdot t \pmod{N} \quad \text{for } t \leftarrow \mathbb{Z}_T, \\ \text{decode}(y) &= (y \pmod{N}) \pmod{2^\ell}, \end{aligned}$$

where  $T = \lfloor 2^{-\ell} \cdot N \rfloor$  and  $\ell < \lfloor \log_2 N \rfloor$ . As there are no modular wrappings, the decoding is always correct. More importantly, Prot. 1 is now secure for small enough  $\ell$ .

**Theorem 3.** *Let  $\pi$  be an additively homomorphic cryptosystem such that the smallest factor of the plaintext order is larger than  $\gamma > 2$ . Then Protocol 1 for transferring  $\ell$ -bit strings is extractable and  $(2^{\ell-1}/\gamma)$ -simulatable.*

*Proof.* Extractability is clear and thus we consider only simulatability. If  $\alpha \neq x$ , then by construction  $y = \text{encode}(\beta) + g$  where  $g$  is chosen uniformly from a non-zero subgroup  $\mathbb{G} \subseteq \mathbb{Z}_N$ . If  $\mathbb{G} = \mathbb{Z}_N$  then  $y$  is uniformly distributed over  $\mathbb{Z}_N$ . Otherwise  $\mathbb{G}$  can be represented as  $p\mathbb{Z}_N$ , where  $p$  is a non-trivial factor of  $N$ , and  $y \pmod{p} \equiv \beta + 2^\ell \cdot t \pmod{p}$ , where  $t \leftarrow \mathbb{Z}_T$  and  $T = \lfloor 2^{-\ell} \cdot N \rfloor$ . Since 2 and  $p$  are relatively prime,  $\{2^\ell \cdot t : t \in \mathbb{Z}_p\} = \mathbb{Z}_p$  and the term  $2^\ell \cdot t \pmod{p}$  covers all elements of  $\mathbb{Z}_p$  almost uniformly. More precisely, the elements of  $\mathbb{Z}_p$  can be divided into two sets:

$$\begin{aligned} \mathcal{T}_0 &= \{c \in \mathbb{Z}_p : \Pr[\beta + 2^\ell \cdot t \pmod{p} = c] = \frac{a}{T}\} & \text{with } |\mathcal{T}_0| &= p - b, \\ \mathcal{T}_1 &= \{c \in \mathbb{Z}_p : \Pr[\beta + 2^\ell \cdot t \pmod{p} = c] = \frac{a+1}{T}\} & \text{with } |\mathcal{T}_1| &= b, \end{aligned}$$

where  $a = \lfloor \frac{T}{p} \rfloor$  and  $b = T - ap$ . Consequently, the statistical difference between  $y \pmod{p}$  and the uniform distribution  $\mathcal{U}(\mathbb{Z}_p)$  can be expressed as

$$\varepsilon = \frac{|\mathcal{T}_0|}{2} \cdot \left( \frac{1}{p} - \frac{a}{T} \right) + \frac{|\mathcal{T}_1|}{2} \cdot \left( \frac{a+1}{T} - \frac{1}{p} \right) = \frac{b(p-b)}{Tp} \leq \frac{p}{4T} \leq \frac{N}{4\gamma T},$$

as  $p(p-b) \leq p^2/4$  and  $p \leq N/\gamma$ . Since  $2^{\ell+1} \leq N$  we get  $T = \lfloor 2^{-\ell} N \rfloor \geq N/2^{\ell+1}$  and thus  $\varepsilon \leq 2^{\ell-1}/\gamma$ . Now note that the distributions  $\text{encode}(\beta) + \mathcal{U}(p\mathbb{Z}_N)$  and  $\mathcal{U}(\mathbb{Z}_N)$

are still  $\varepsilon$ -close, as we can express

$$\Pr [\text{encode}(\beta) + \mathcal{U}(p\mathbb{Z}_N) = c \pmod{N}] = \frac{p}{N} \cdot \Pr [\text{encode}(\beta) = c \pmod{p}] .$$

Hence, we can use a simulator  $\text{Sim}_{\text{sk}}^*(\mathbf{q}, f^*)$  that outputs  $\text{Enc}_{\text{pk}}(\text{encode}(\beta))$  if  $f^* = \beta$  and  $\text{Enc}_{\text{pk}}(m)$  for  $m \leftarrow \mathbb{Z}_N$  otherwise.  $\square$

**Corollary 1.** *Let  $\pi$  be an  $(\tau, \varepsilon_1)$ -IND-CPA-secure additively homomorphic cryptosystem such that the smallest factor of the plaintext order is larger than  $\gamma > 2$ . Then Protocol 1 for transferring  $\ell$ -bit strings is  $(\tau, \varepsilon_1; \varepsilon_2)$ -relaxed-secure for  $\varepsilon_2 = 2^{\ell-1}/\gamma$ .*

**The Maximal Throughput of DIE Protocol.** First, note that if we want to achieve  $\varepsilon$ -server-privacy then we must choose  $\ell = \lfloor \log_2(2\varepsilon\gamma) \rfloor$ , where  $\gamma$  is the lower bound to non-trivial factors of  $N$ . Usually, it is sufficient to take  $\varepsilon = 2^{-80}$  and thus  $N$  cannot have smaller factors than  $2^{80}$  if the server wants to release Boolean secrets. For the Paillier cryptosystem the smallest factor of  $N$  is approximately  $\sqrt{N}$ , and consequently, one can transfer  $\ell = \lfloor \log_2(2\sqrt{N}\varepsilon) \rfloor \approx 0.5 \log_2 N + \log_2 \varepsilon$  bits. For standard 1024-bit RSA modulus and  $\varepsilon = 2^{-80}$ , one can take  $\ell = 433$ .

As our DIE protocol is extractable and simulatable, a forked composition of  $t$  protocols enables transfer of a  $t\ell$ -bit secret, where the achieved server-privacy is  $\frac{|\beta|}{\ell\gamma} \cdot 2^{\ell-1}$ . Smaller values of  $\ell$  increase the maximal length of  $\beta$  but also decrease the ratio between the desired communication  $|\beta|$  and the total communication  $|\mathbf{q}| + |\mathbf{a}|$  and make the protocol less efficient. In other words, a bad encoding  $\text{encode}(\beta)$  with a small capacity can significantly decrease efficiency. As our target distribution is  $\mathcal{U}(\mathbb{Z}_N)$  then it is straightforward to derive entropy bounds for the capacity:  $H(\mathbb{Z}_N) \approx H(\text{encode}(\beta) + p\mathbb{Z}_N) \leq H(\text{encode}(\beta)) + H(p\mathbb{Z}_N) \leq \log_2 |\text{encode}(\beta)| + H(p\mathbb{Z}_N)$ , where  $|\text{encode}(\beta)|$  denotes the size of the support. As the encoding must be uniquely decodable, the capacity of a single reply  $\ell \leq \log_2 \frac{N}{|\text{encode}(\beta)|} \lesssim \min_p H(p\mathbb{Z}_n) = \log_2 \Phi$ , where  $\Phi$  is the smallest prime factor of  $N$ . Thus, the encoding is optimal up to a constant additive term  $\log_2 \varepsilon$ . The result can be generalised for any target distribution using a more detailed analysis.

## 5 Generic Construction for Conditional Disclosure of Secrets

Many protocols are secure only if client submits inputs  $\alpha$  from a limited range  $\mathcal{S}$ . Cleverly chosen  $\alpha \notin \mathcal{S}$  can either partially or completely reveal the server's input  $\beta$ . Therefore, the server must somehow verify that  $\alpha \in \mathcal{S}$ . Classically, this is done by a zero-knowledge proof that  $\text{Dec}_{\text{sk}}(c) \in \mathcal{S}$ . However, this either increases the number of messages or requires a security model with a common reference string or random oracles. A conditional disclosure of secrets (CDS) reaches the same goal without extra messages and exotic assumptions. In a CDS protocol, the client should learn a secret  $\beta$  only if  $\text{Dec}_{\text{sk}}(\mathbf{q}) \in \mathcal{S}$ , where the query vector  $\mathbf{q}$  consists of ciphertexts  $\text{Enc}_{\text{pk}}(\alpha_1), \dots, \text{Enc}_{\text{pk}}(\alpha_m)$  and the set  $\mathcal{S}$  is public. Since the server can use  $\beta$  as a one-time pad to encrypt the original reply  $\mathbf{a}$ , the client learns nothing about the outputs of the original protocol if  $\alpha \notin \mathcal{S}$  and the modified protocol becomes server-private.

A CDS protocol can be straightforwardly constructed as a forked composition of individual DIE protocols for  $\{\text{Dec}_{\text{sk}}(c) = x\}_{x \in \mathcal{S}}$  that share the same secret  $\beta$  but such composition is inefficient. Therefore, we show how to use Benaloh-Leichter secret sharing scheme [BL88] together with slightly extended DIE protocols to achieve a more computation and communication efficient CDS protocol (*circuit CDS*).

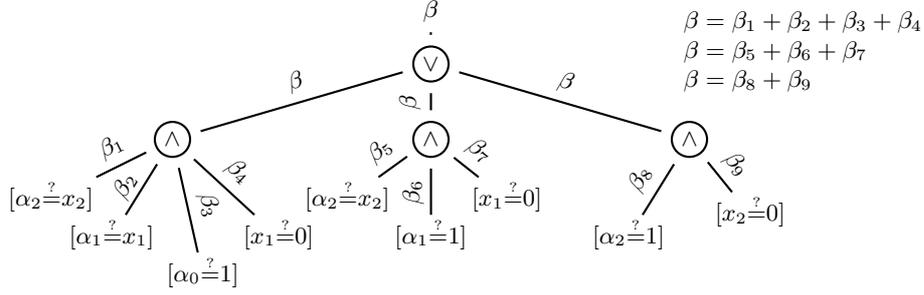
**Conjunctive affine zero tests.** First, we present an optimisation for specific sets. Recall that our DIE protocol is secure since  $\text{encode}(\beta) + \mathcal{U}(\mathbb{G}) \stackrel{\epsilon}{\sim} \mathcal{U}(\mathbb{Z}_N)$  if  $\mathbb{G} \neq \{0\}$ . Similarly, we can construct CDS protocols for *conjunctive affine zero tests*  $\Psi_0(\alpha) = \bigwedge_{j=1}^v [\sum_{i=1}^m s_{ij} \alpha_i \stackrel{?}{=} x_j]$ , where  $\{x_i\}$  and  $\{s_{ij}\}$  are public constants:

1. The client sends  $\mathbf{q} = (c_1, \dots, c_m)$  where  $c_i = \text{Enc}_{\text{pk}}(\alpha_i)$ .
2. The server halts if some  $c_1, \dots, c_m$  is not a valid ciphertext, otherwise it replies  $\mathbf{a} = \prod_{j=1}^v (\prod_{i=1}^m c_i^{s_{ij}} \cdot \text{Enc}_{\text{pk}}(-x_j))^{r_j} \cdot \text{Enc}_{\text{pk}}(\text{encode}(\beta))$  for  $r_1, \dots, r_v \leftarrow \mathbb{Z}_N$ .
3. The client restores  $y = \text{Dec}_{\text{sk}}(\mathbf{a})$  and outputs  $\text{decode}(y)$ .

As  $y = \sum_{j=1}^v (\sum_{i=1}^m \alpha_i s_{ij} - x_j) r_j + \text{encode}(\beta) = \text{encode}(\beta) + \mathbb{G}_1 + \dots + \mathbb{G}_v$ , then  $y = \text{encode}(\beta) + \mathcal{U}(\mathbb{G})$  for a non-zero sub-group  $\mathbb{G}$  if some zero-tests do not hold. The latter follows from the fact that  $r_1, \dots, r_v$  are independently chosen. Hence, the claims of Thm. 3 hold also for the CDS protocol given above. Of course, when the plaintext order is prime then there is no need to use probabilistic encoding and we can use the construction given in [AIR01]. Notably, such simplified construction has been used in [BGN05] together with a cryptosystem that has a composite plaintext order. Paradoxically, the latter construction is still computationally secure, as the client must compute arbitrary discrete logarithms to recover a coset  $\beta + \mathbb{G}$ .

**Circuit CDS protocol.** For any set  $\mathcal{S}$ , we can write the predicate  $\Psi_{\mathcal{S}}(\alpha) := [\alpha \in \mathcal{S}]$  as a monotonous combination of affine zero tests, i.e., the formula consists of Boolean operations  $\wedge$  and  $\vee$  together with atomic terms  $\Psi_0(\alpha) = \bigwedge_{j=1}^v [\sum_{i=1}^m s_{ij} \alpha_i \stackrel{?}{=} x_j]$ . For efficiency reasons, we might express the input  $\alpha$  as a bit-vector. The server can later use properties of additively homomorphic encryption to restore the original ciphertexts.

First, the server uses the Benaloh-Leichter secret sharing scheme to assign sub-secrets  $\beta_i$  to each leaf test  $\Psi_0(\alpha)$  so that the client can reconstruct the secret  $\beta \in \{0, 1\}^\ell$  if  $\Psi(\alpha)$  holds and the secrets of true leaves are revealed. Fig. 1 illustrates how secret  $\beta$  is propagated through the circuit of  $\Psi(\alpha) = [\alpha > x]$  without optimisation. Namely, the master secret  $\beta$  is assigned to the topmost gate of the circuit. For every  $\vee$ -gate, the output secret is just pushed downwards. For every  $\wedge$ -gate  $\psi$  with  $u$  children and a secret  $\beta_\psi$  assigned to it, sub-secrets  $\beta_1, \dots, \beta_{u-1} \leftarrow \{0, 1\}^\ell$  and  $\beta_u \leftarrow \beta_\psi - \beta_1 - \dots - \beta_{u-1} \pmod{2^\ell}$  are assigned to the children. One can also use threshold operations:  $\text{THR}_v(x_1, \dots, x_s) = 0$  if and only if at least  $v$  values  $x_j$  are equal to 1. For a  $\text{THR}_v$  gate, generate a random  $(v-1)$ -degree polynomial  $f_\psi$  with  $f_\psi(0) = \beta_\psi$  and assign the secret  $f_\psi(i)$  to its  $i$ th child. Finally, the server uses a forked composition of CDS protocols for leaf tests  $\Psi_0$  to release sub-secrets associated to each leaf. The client recomputes the secret from leaf values by inversely following the secret generation.



**Fig. 1.** An unoptimised circuit for  $\Psi(\alpha) = [\alpha > x]$  where secrets are pushed down to DIE leaves. The circuit can be further optimised by replacing  $\wedge$ -gates with conjunctive affine equality tests

**Theorem 4.** *If the leaf CDS protocol is extractable and  $\varepsilon_2$ -simulatable, then the circuit CDS protocol for  $\Psi_S$  is extractable and  $\mathcal{L}(\Psi_S) \cdot \varepsilon_2$ -simulatable, where  $\mathcal{L}(\Psi_S)$  is the number of leaves. If the cryptosystem is  $(\tau, \varepsilon_1)$ -IND-CPA secure and  $\mathfrak{q}$  consists of  $m$  ciphertexts, then the protocol is  $(\tau - \mathcal{O}(1), m\varepsilon_1; \mathcal{L}(\Psi_S) \cdot \varepsilon_2)$ -relaxed-secure.*

*Proof.* Given the main secret  $\beta$  it is straightforward to reconstruct the leaf-level secrets. Otherwise, if  $\Psi_S(\alpha) = 0$  then the sub-secrets  $\beta_i$  that are assigned to true atoms  $\Psi_0(\alpha) = 1$  are independent and are uniformly distributed. Hence, a world with  $\mathcal{L}(\Psi_S)$  ideally implemented leaf CDS protocols can be perfectly simulated in the world where  $\beta$  is released only if  $\alpha \in \mathcal{S}$ . Now, the simulatability follows directly from Thm. 2. The second claim follows from Thm. 1 and the basic properties of IND-CPA encryption.  $\square$

If the CDS protocol is based on the new DIE protocol, then we can estimate how many bits are needed to transfer  $\ell$ -bit secrets. For the 1024-bit Paillier cryptosystem and  $2^{-80}$ -sever-privacy, a single ciphertext can fit 393 bits provided that the corresponding circuit has less than  $2^{40}$  leaves; the message expansion is roughly  $|\mathfrak{a}|/\ell \approx 5.2 \cdot \mathcal{L}(\Psi)$ .

As negations can be expressed by conjunctive affine zero tests, then they can appear only in the leaf level, i.e., the formula  $\Psi(\alpha)$  must be in a negation normal form (NNF). Many practically interesting sets have compact NNF-s, but for some circuits  $\Psi$  such normal form is exponentially larger. We can circumvent the problem by using auxiliary inputs  $w$ . Consider the circuit representation of  $\Psi$  that consists of unary  $\neg$ -gates and binary  $\wedge$ - and  $\vee$ -gates. Denote all output wires of logical gates by auxiliary labels  $w_i$ . Now, we can represent assignments  $w_u \leftarrow w_s \wedge w_t$  and  $w_u \leftarrow w_s \vee w_t$  with the formulae

$$\begin{aligned} [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \wedge [w_t \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \vee [w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \ . \\ [w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \wedge [w_t \stackrel{?}{=} 0] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \ , \end{aligned}$$

and  $w_u \leftarrow \neg w_s$  as  $[w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 0]$ . Therefore, we can in principle construct a new formula  $\overline{\Psi}(\alpha, w)$  in NNF such that  $\Psi(\alpha) = 1 \iff \exists w : \overline{\Psi}(\alpha, w) = 1$  and the size of  $\overline{\Psi}(\alpha, w)$  is proportional to the gate count of  $\Psi(\alpha)$ . Consequently, we can always construct efficient circuit CDS protocols for efficiently recognisable sets.

**Query phase:**

The client sends  $q = (c_1, \dots, c_m)$  to the server, where  $c_i \leftarrow \text{Enc}_{\text{pk}}(\alpha_i)$  for  $i \in \{1, \dots, m\}$ .

**Transfer phase:**

The server computes the reply  $\mathbf{a} = (d_1, \dots, d_n)$  according to the original protocol  $\Pi$

The server applies one-time pad  $e_i \leftarrow d_i \cdot \text{Enc}_{\text{pk}}(t_i)$  for  $t_i \leftarrow \mathbb{Z}_N$  and  $i \in \{1, \dots, m\}$ .

The server computes the CDS reply  $\mathbf{a}_{\text{cds}}$  for  $\text{Dec}_{\text{sk}}(q) \in \mathcal{S}$  with secret  $\beta = t_1 \parallel \dots \parallel t_m$ .

The server replies  $(e_1, \dots, e_n)$  and  $\mathbf{a}_{\text{cds}}$ .

**Post-processing:**

The client recovers the secrets  $t_i$  from  $\mathbf{a}_{\text{cds}}$  and computes  $\hat{d}_i \leftarrow e_i \cdot \text{Enc}_{\text{pk}}(-t_i)$ .

Next, the client proceeds with the original protocol  $\Pi$ .

**Protocol 2:** CDS transformation for additively homomorphic two-message protocols

**CDS Transformation.** It is straightforward to use a CDS protocol to transform any additively homomorphic two-message protocol that is secure in the semihonest model to a modified two-message protocol that is relaxed-secure. Let the query  $q$  consist of  $m$  ciphertexts  $(c_1, \dots, c_m)$ . Protocol  $\Pi$  is secure in the semihonest model, when there exist a set of valid inputs  $\mathcal{S}$  such that the client learns only  $f(\alpha, \beta)$ , provided that  $\text{Dec}_{\text{sk}}(q) = (\alpha_1, \dots, \alpha_m) \in \mathcal{S}$ . Let us use a sufficiently long secret  $\beta$  as a one-time pad to decrypt the original reply  $\mathbf{a}$  and release the secret only if  $\text{Dec}_{\text{sk}}(q) \in \mathcal{S}$ . Then the corresponding protocol is clearly relaxed-secure. In many cases, the reply  $\mathbf{a}$  consists of re-randomised ciphertexts and we can reduce the length of the secret  $\beta$ , see Prot. 2.

**Theorem 5.** *If the two-message additively homomorphic protocol  $\Pi$  is correct,  $(\tau, \varepsilon_1)$ -client-private and  $\varepsilon_2$ -simulatable for  $\alpha \in \mathcal{S}$  and the CDS protocol for the set  $\mathcal{S}$  is  $\varepsilon_3$ -simulatable, then Protocol 2 is correct and  $(\tau, \varepsilon_1; \max\{\varepsilon_2, \varepsilon_3\})$ -relaxed-secure.*

*Proof.* Due to the re-randomisation, the recovered replies  $\hat{d}_i$  have the same distribution as  $d_i$ , thus correctness is evident. Client-privacy is evident as both protocols share the query  $q$ . For server-privacy, note that if  $\alpha \in \mathcal{S}$ , we can first use the original simulator to simulate  $d_i$  and then apply the CDS transformation to the simulation output. The corresponding simulation is  $\varepsilon_2$ -close to the real run, since the original reply is not more than  $\varepsilon_2$  away from the simulated one. Otherwise,  $(e_1, \dots, e_n)$  are random ciphertexts and thus perfectly simulatable. Now if we add a simulated CDS reply  $\hat{\mathbf{a}}_{\text{cds}}$ , then the aggregated reply  $\hat{\mathbf{a}}_{\text{cds}}, e_1, \dots, e_n$  is  $\varepsilon_3$ -close to the real protocol transcript, as the CDS is  $\varepsilon_3$ -simulatable. The claim follows, as  $\text{Dec}_{\text{sk}}(q)$  is either in  $\mathcal{S}$  or not.  $\square$

**Optimisations.** If all replied ciphertexts of the original protocol are in the fixed range, i.e.,  $\text{Dec}_{\text{sk}}(d_i) \in \{0, 1\}^\ell$  then full recovery of  $t_i$  is not necessary. It is sufficient to send  $t_i \bmod 2^\ell$  together with a extra bit needed to indicate a possible wrapping  $t_i \geq N - 2^\ell$  and the message expansion rate can be less than  $\mathcal{L}(\Psi)$ . Secondly, note that the communication overhead of the CDS transformation is linear in  $|\mathbf{a}|$ . Therefore, the transformation is quite inefficient when  $|\mathbf{a}|$  is long. To get better performance, the server can use symmetric encryption to garble the original reply and a CDS protocol to release the corresponding key. The output is still computationally simulatable and thus we achieve

computational server-privacy. A block cipher in counter mode is the best encryption method, as then the client can efficiently decrypt only necessary parts of  $\mathbf{a}$ .

## 6 Practical Applications of Crypto-Computing Techniques

In this section, we show how to use additively homomorphic two-message protocols to solve several important cryptographic tasks. Here, the query  $\mathbf{q}$  is a vector of ciphertexts, and the reply is computed by combining the identities (1) and (2) with Prot. 1. Note that the outputs of crypto-computed sums and products are perfectly simulatable provided that the end result is re-randomised. Consequently, client-privacy follows form  $(\tau, \varepsilon_1)$ -IND-CPA security and server-privacy follows from the basic properties of forked composition, see Thm. 1 and 2. Shortly put, the resulting protocols are  $(\tau - \mathcal{O}(1), m\varepsilon_1; n\varepsilon_2)$ -relaxed-secure, where  $m$  is the number of ciphertexts and  $n$  is the number of DIE instances, provided that the basic DIE protocol is  $\varepsilon_2$ -simulatable.

Sometimes we must also prove that knowledge of  $f(\alpha, \beta)$  is equivalent to the knowledge of  $f_1(\alpha, \beta), \dots, f_s(\alpha, \beta)$ , i.e., design a protocol for  $f$  based on generic operations. As for 1024-bit Paillier and  $2^{-80}$ -server-privacy, we can transfer 393 bits in the individual DIE reply whenever the number of DIE instances is less than  $2^{40}$ , the resulting protocols are really efficient.

**Oblivious Transfer.** Recall that a 1-out-of- $n$  oblivious transfer (OT) protocol implements an ideal functionality  $f(\alpha; \beta_1, \dots, \beta_n) = \beta_\alpha$  if  $\alpha \in \{1, \dots, n\}$  and  $\perp$  otherwise. Already in [AIR01], the authors showed that such a protocol can be expressed as a forked composition of  $n$  individual DIE protocols:

- release  $\beta_1$  if  $\alpha = 1$ ,
- $\vdots$
- release  $\beta_n$  if  $\alpha = n$ .

Therefore, we get a relaxed-secure implementation of oblivious transfer by using Prot. 1 to implement all instances of DIE protocols. Moreover, a client can use any CPIR protocol to obliviously choose the  $\alpha$ th reply of the DIE. Hence, we have just described a generic transformation from any CPIR to a relaxed-secure oblivious transfer.

An alternative approach was taken by Chang [Cha04] who proved that the basic DIE protocol from [AIR01] leaks at most  $\beta_{\alpha_1} \bmod p_1$  and  $\beta_{\alpha_2} \bmod p_2$  whenever the plaintext order is a product of two primes  $p_1$  and  $p_2$ . In the corresponding 1-out-of- $n$  OT protocol an honest client has to encrypt values that depend on the secret key and thus the client-privacy does not follow directly from IND-CPA security.

**Millionaire’s protocol with logarithmic communication.** The millionaire’s problem is: given client’s private input  $\alpha$  and server’s private input  $x$ , decide whether  $\alpha > x$ . Although numerous solutions have been proposed for this problem, none of the proposals is completely satisfactory. For example, the two-message protocol of Blake and Kolesnikov [BK04] is server-secure only in the semihonest model since encrypted inputs must be in correct range, it can leak information otherwise. To solve that type of

problems, consider a circuit CDS protocol for a public set  $\mathcal{S}_x = \{\alpha \in \{0, 1\}^m : \alpha > x\}$ . Writing  $\alpha$  bit by bit  $(\alpha_{m-1}, \dots, \alpha_0)$ , we obtain

$$\begin{aligned} \Psi_{\mathcal{S}_x}(\alpha) = & ([\alpha_{m-1} \stackrel{?}{=} 1] \wedge [x_{m-1} \stackrel{?}{=} 0]) \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} 1] \wedge [x_{m-2} \stackrel{?}{=} 0]) \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} x_{m-2}] \wedge [\alpha_{m-3} \stackrel{?}{=} 1] \wedge [x_{m-3} \stackrel{?}{=} 0]) \vee \dots \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} x_{m-2}] \wedge \dots \wedge [\alpha_1 \stackrel{?}{=} x_1] \wedge [\alpha_0 \stackrel{?}{=} 1] \wedge [x_0 \stackrel{?}{=} 0]) . \end{aligned}$$

Here, every row corresponds to one conjunctive affine equality test. Fig. 1 depicts the corresponding unoptimised circuit. Now consider the modified protocol where  $\beta_0$  is a publicly fixed  $\ell$ -bit secret and the server randomly reorders the leaf CDS replies  $\alpha_1, \dots, \alpha_m$ . Finally, the client outputs 1 if one of the recovered CDS outputs is  $\beta_0$ . As the formula  $\Psi_{\mathcal{S}_x}(\alpha)$  is a disjunction of affine zero tests, then in the ideal world, the client learns a randomly shuffled set  $\{\beta_0, \perp, \dots, \perp\}$  if  $\alpha > x$  and  $\{\perp, \perp, \dots, \perp\}$  otherwise. Hence, the modified protocol is server-private even if  $x$  is private and we have obtained a relaxed-secure solution to the millionaire problem that fails with probability  $2^{-\ell}$ . The total communication of our solution is  $2m$  ciphertexts, the client's computation is  $\Theta(m)$  and the server's computation is  $\Theta(m^2)$ , and we only assume that the underlying additively homomorphic cryptosystem is IND-CPA secure. The server's workload can be reduced  $\Theta(m)$  as in the Blake-Kolesnikov protocol, if we first compute a recursion  $t_i = (\alpha_i - x_i)r_i + \dots + (\alpha_{m-1} - x_{m-1})r_{m-1}$  for  $r_i \leftarrow \mathbb{Z}_N$  and then re-randomise it by crypto-computing  $u_i = t_i s_i$  for  $s_i \leftarrow \mathbb{Z}_N$ .

Interestingly enough, one can view our solution as an efficient generalisation of the Fischlin protocol [Fis01]. The latter can be alternatively described as a CDS protocol based on additively homomorphic cryptosystem over  $\mathbb{Z}_2$ . Due to the small message space, the Fischlin's protocol requires a parallel run of  $\ell$  protocols to achieve the same reliability as our protocol, i.e., our protocol is  $\ell$  times more efficient.

**Conditional OT.** In a *conditional oblivious transfer* protocol for public predicate  $\Psi$ , the client has a private input  $\alpha$  and the server has a private input  $(x, \beta_0, \beta_1)$ . The client obtains  $\beta_1$  if  $\Psi(\alpha, x) = 1$  and  $\beta_0$  otherwise. Assume that the master secret  $\beta$  is reconstructed identically for the circuits without witnesses  $\Psi$  and  $\neg\Psi$  and the reconstruction process and the number of true leaves leaks nothing about  $x$  except  $\Psi(\alpha, x)$ . In particular, assume that the master secret can be reconstructed from randomly shuffled shares. Let  $\mathcal{B}_{\Psi(\alpha, x)}$  and  $\mathcal{B}_{\neg\Psi(\alpha, x)}$  be the shuffled CDS replies in the ideal world. Then given a shuffled set of sets  $\{\mathcal{B}_{\Psi(\alpha, x)}, \mathcal{B}_{\neg\Psi(\alpha, x)}\}$ , one can learn only  $\beta_{\Psi(\alpha, x)}$  and nothing more, provided that the number of leaf tests is equal  $|\mathcal{B}_{\Psi(\alpha, x)}| = |\mathcal{B}_{\neg\Psi(\alpha, x)}|$ .

This leads to the following COT protocol. First, the server assigns  $\beta_0$  to  $\neg\Psi$  and  $\beta_1$  to  $\Psi$  and adds trailing zeroes to leaf secrets of one circuit and trailing ones to the remaining sub-secrets. Next, the server constructs replies for each leaf CDS and sends randomly shuffled replies back. Finally, the client restores sets  $\mathcal{B}_{\Psi(\alpha, x)}$  and  $\mathcal{B}_{\neg\Psi(\alpha, x)}$  and reconstructs  $\beta_{\Psi(\alpha, x)}$ . The failure probability is bounded by  $2^{-k} \cdot \mathcal{L}(\Psi)$  where  $k$  is the number of trailing zeroes and ones. Since  $[\alpha > x]$  and  $[\alpha \leq x]$  have such symmetrical circuits, we can construct a COT protocol for  $[\alpha > x]$  and for many other relations.

**Electronic voting and auctions without random oracles.** E-voting and auction protocols based on homomorphic encryption [CGS97,DJ01,LAN02] are natural extensions of homomorphic two-message protocols, since the secret key is known by the election tallier (or a coalition of talliers) to whom the server forwards the second message. In such protocols, conditional disclosure of secrets can be used to guarantee security of the election authority against malicious voters and a semihonest server. As in [BGN05], consider an electronic voting protocol where every voter sends an encryption  $c_i \leftarrow \text{Enc}_{pk}(v_i)$  to talliers. We assume that the protocol is secure if  $v_i \in \mathcal{S}$  for some publicly known set  $\mathcal{S}$ ; this is true in typical e-voting protocols [CGS97,DJ01].

In the existing protocols, it is usually assumed that every voter accompanies his or her vote with a non-interactive zero-knowledge proof that  $v_i \in \mathcal{S}$ . Instead, the talliers can jointly apply the CDS protocol, with output secret 0, to  $c_i$  (this can be done very efficiently if  $\mathcal{S}$  is the set of powers of a fixed integer) and then threshold-decrypt the result. If the plaintext is equal to 0, talliers accept the vote as correct. Of course, every step of the talliers has to be accompanied by a zero-knowledge proof of correctness (to each other and to every possible outside observer), but since the number of talliers is significantly smaller than the number of voters, this is doable in practise, see [BGN05].

As the result, we get a voter-private, universally verifiable and robust e-voting scheme where the voters only have to perform one encryption, assuming only that there exists an IND-CPA secure additively homomorphic public-key cryptosystem. The same trick can be used to eliminate the need for random oracles in a similar electronic auction scheme of [LAN02] and in many other similar protocols. Compared to the protocols of [BGN05], our protocols are more efficient since they are based on genuine additive homomorphic cryptosystem whereas [BGN05] uses a lifted version of ElGamal and thus there one has to compute discrete logarithms. Moreover, their cryptosystem is secure under less established security assumptions.

**Multiplicative relations and polynomial arithmetic** Finally, we illustrate the power of using auxiliary witnesses. It is well known that multiplicative relation  $[z \stackrel{?}{=} xy]$  does not have a compact NNF. However, we can still construct efficient circuit CDS protocol by introducing a suitable witness  $w$ . Let  $x, y \in \{0, 1\}^m$  and  $z \in \{0, 1\}^{2m}$  be sent to the server by individually encrypting each bit of  $x, y, z$  and let  $w_0, \dots, w_{m-1}$  be auxiliary variables such that  $w_i = xy_i$ . Then  $xy = w_0 + 2w_1 + \dots + 2^{m-1}w_{m-1}$  and the formula  $\Psi_{[z=xy]}$  can be expressed as a conjunction of tests: (1)  $x_{m-1}, \dots, x_0 \in \{0, 1\}$ , (2)  $[y_i \stackrel{?}{=} 0] \wedge [w_i \stackrel{?}{=} 0] \vee [y_i \stackrel{?}{=} 1] \wedge [w_i \stackrel{?}{=} x]$  for  $i \in \{0, \dots, m-1\}$  and  $x$  is crypto-computed as  $x_0 + \dots + 2^{m-1}x_{m-1}$ , and (3)  $[z \stackrel{?}{=} w_0 + \dots + 2^{m-1}w_{m-1}]$ .

Several papers, see e.g. [KS05], use additively homomorphic two-message protocols in a setting where one encrypts the coefficients of some polynomials, where the important quantity is the set of roots of this polynomial. For example, if  $F_1$  is the set of roots of  $f_1(x)$  and  $F_2$  is the set of roots of  $f_2(x)$  then  $F_1 \cup F_2$  is the set of roots of  $f_1(x) \cdot f_2(x)$ . Consequently, we can also construct a CDS protocol for the set to prove that  $g(x) = f_1(x) \cdot f_2(x)$ , as the  $i$ th coefficient  $g_i = f_{10}f_{2i} + \dots + f_{1i}f_{20}$ . Now, we can also verify that for some sets  $F_1, F_2$  and  $G$ , it holds that  $F_1 \cup F_2 = G$ .

## 7 Theoretical Implications

Although we stated our results in the PKI model, where a trusted key generator generates a key pair  $(sk, pk) \leftarrow \text{Gen}$  and privately transfers  $(sk, pk)$  to the client and  $pk$  to the server, they can be easily implemented in the standard model. Namely, we can eliminate the PKI assumption if the client executes once, separately and in an isolated manner (that is, no other messages of different protocols are sent by the client at the same time), with every server a zero-knowledge proof of knowledge that  $pk$  is valid and that he knows the corresponding secret key. This is followed by the real protocol. In the security proof, the simulator extracts the secret key by rewinding and thereafter continues to work as previously. Since we require statistical server-security—and thus can use an unbounded simulator—then it is actually sufficient to have a zero-knowledge proof that the key is correct: the simulator just computes the secret key corresponding to the (correct) public key. It is even irrelevant whether the client computes the public key with a correct distribution, since for the proof we only need the existence of the secret key. Therefore, the amortised message complexity is still two-messages in the standard model, as the verification of a public key must be carried out only once.

It is well known that secure two-party protocols require at least three messages, therefore, it is impossible to obtain full security of two-message protocols in the malicious model. In fact, one cannot achieve more than relaxed-security in two messages even in the PKI model. Consequently, the CDS-transformation presented in Sect. 5 is a universal round-optimal transformation from semihonest model to relaxed-secure model whenever the first message contains only ciphertexts. Moreover, computational and communication resources are linear in the size of the circuit that is needed to test a validity of an input. More formally, assume that for sets  $\mathcal{S}_m$  of  $m$ -bit strings exists a polynomial-size formula  $\Psi(\alpha, w)$  such that  $\alpha \in \mathcal{S}_m$  iff  $\exists w : \Psi(\alpha, w) = 1$ . Then there exists also a polynomial-size formula  $\bar{\Psi}(\alpha, \bar{w})$  in a negation normal form such that  $\alpha \in \mathcal{S}_m$  iff  $\exists \bar{w} : \bar{\Psi}(\alpha, \bar{w}) = 1$ . Therefore, there exist a family of polynomial-time CDS protocols for an arbitrary set  $\mathcal{S}$  in  $\text{NP/poly}$ . Such protocols can be automatically generated in polynomial time for every set  $\mathcal{S}$  that can be described by any  $\text{NP}$  relation.

Alternative classical round-preserving methods that guard against malicious clients are based on non-interactive zero-knowledge proofs, i.e., we have to either rely on random oracles or use the *common reference string* (CRS) model. While CRS is a plausible model for protocol design, constructing efficient non-interactive zero-knowledge protocols for  $\text{NP}$  in the CRS model has been a long-standing open problem. Thus, our result is also appealing from the complexity-theoretical viewpoint.

As stated already in Sect. 6, the DIE-based OT protocol leads to a general transformation from CPIR to information-theoretically server-private OT, as the client can use the CPIR protocol to fetch only the answer of the  $\alpha$ th DIE protocol. In particular, there exists a generic CPIR construction for any IND-CPA secure additively homomorphic cryptosystem [Ste98] with sublinear-but-superpolylogarithmic communication. Therefore, there exists also an OT protocol with comparable communication under the sole assumption that IND-CPA secure additively homomorphic cryptosystems exist. Under the assumption that IND-CPA secure length-flexible additively homomorphic cryptosystem exist, one can construct a CPIR protocol [Lip05] with communication  $\Theta(k \cdot \log^2 n + \ell \cdot \log n)$  where  $k$  is the security parameter. Consequently, we

can construct an OT with communication  $\Theta(k \cdot \log^2 n + \ell \cdot \log n)$ , if an IND-CPA secure length-flexible additively homomorphic cryptosystem exists. Finally due to the results of Gentry and Ramzan [GR05], there also exists an OT protocol with optimal communication  $\Theta(\log n + \ell + k)$ , if we assume that  $\Phi$ -Hiding is hard and that an IND-CPA secure additively homomorphic cryptosystem exists.

Another two-message OT protocol was proposed by Kalai [Kal05]. Her protocol is secure in the standard model, whereas our protocol requires a zero-knowledge proof that the public key is valid. On the other hand, the query of Kalai's protocol does not consist of ciphertexts and thus cannot be used for the CDS protocol. Moreover, Thm. 3 holds even with incorrectly formed pk provided that the corresponding encryption rule is additively homomorphic and it is still possible to detect invalid ciphertexts. Therefore, we can omit the zero-knowledge proofs for pk provided that we can verify that the plaintext order does not have too small factors. For small enough  $\gamma$  and public plaintext order this can be done efficiently by using Lenstra's Elliptic Curve Method, see App. A for further details. Hence, it is possible to achieve two messages as non-amortised round-complexity in the standard model under stronger computational assumptions.

Finally, note that small detectable factors of  $N$  can be effectively eliminated. Namely, a server can eliminate a known factor  $p$  by multiplying a ciphertext  $\text{Enc}_{\text{pk}}(x)$  with  $\text{Enc}_{\text{pk}}(pr)$  for  $r \leftarrow \mathbb{Z}_N$ . Then the client can learn only a coset  $x + p\mathbb{Z}_N$ , i.e., we have established a new cryptosystem over a new message space  $\mathbb{Z}_N/p\mathbb{Z}_N \simeq \mathbb{Z}_N/p$ .

**Acknowledgements.** We would like to thank Phil Carmody, Yuval Ishai and Vladimir Kolesnikov for useful comments. The work was partially supported by the Finnish Academy of Sciences and by the Estonian Science Foundation, grant 6848.

## References

- [AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.
- [BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF Formulas on Ciphertexts. In *The Second Theory of Cryptography Conference, TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, 2005. Springer Verlag.
- [BK04] Ian F. Blake and Vladimir Kolesnikov. Strong Conditional Oblivious Transfer and Computing on Intervals. In *Advances on Cryptology — ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.
- [BL88] Josh Benaloh and Jerry Leichter. Generalized Secret Sharing and Monotone Functions. In *Advances in Cryptology—CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, 1988. Springer-Verlag, 1990.
- [CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, 1997. Springer-Verlag.
- [Cha04] Yan-Cheng Chang. Single Database Private Information Retrieval with Logarithmic Communication. In *The 9th Australasian Conference on Information Security and Privacy (ACISP 2004)*, volume 3108 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.

- [DJ01] Ivan Damgård and Mads Jurik. A Generalisation, a Simplification and Some Applications of Paillier’s Probabilistic Public-Key System. In *Public Key Cryptography 2001*, volume 1992 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.
- [Elg85] Taher Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4), 1985.
- [Fis01] Marc Fischlin. A Cost-Effective Pay-Per-Multiplication Comparison Method for Millionaires. In *Topics in Cryptology - CT-RSA 2001, The Cryptographer’s Track at RSA Conference 2001*, volume 2020 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.
- [FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.
- [GIKM00] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting Data Privacy in Private Information Retrieval Schemes. *Journal of Computer and System Sciences*, 60(3), June 2000.
- [GLLM04] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On Private Scalar Product Computation for Privacy-Preserving Data Mining. In *Information Security and Cryptology - ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.
- [GM82] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, 1982. ACM.
- [GR05] Craig Gentry and Zulfikar Ramzan. Single-Database Private Information Retrieval with Constant Communication Rate. In *The 32nd International Colloquium on Automata, Languages and Programming, ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.
- [Kal05] Yael Tauman Kalai. Smooth Projective Hashing and Two-Message Oblivious Transfer. In *Advances in Cryptology — EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.
- [KS05] Lea Kissner and Dawn Song. Privacy-Preserving Set Operations. In *Advances in Cryptology — CRYPTO 2005, 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.
- [LAN02] Helger Lipmaa, N. Asokan, and Valteri Niemi. Secure Vickrey Auctions without Threshold Trust. In *Financial Cryptography — Sixth International Conference*, volume 2357 of *Lecture Notes in Computer Science*, 2002. Springer-Verlag.
- [Len87] Hendrik W. Lenstra, Jr. Factoring integers with Elliptic Curves. *Annals of Mathematics*, 126(2), 1987.
- [Lip05] Helger Lipmaa. An Oblivious Transfer Protocol with Log-Squared Communication. In *The 8th Information Security Conference (ISC’05)*, volume 3650 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.
- [LLM05] Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. Private Itemset Support Counting. In *Information and Communications Security, 7th International Conference, ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.
- [NP99] Moni Naor and Benny Pinkas. Oblivious Transfer and Polynomial Evaluation. In *Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing*, 1999. ACM Press.
- [Pai99] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Advances in Cryptology — EUROCRYPT ’99*, volume 1592 of *Lecture Notes in Computer Science*, 1999. Springer-Verlag.

- [Ste98] Julien P. Stern. A New and Efficient All or Nothing Disclosure of Secrets Protocol. In *Advances on Cryptology — ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, 1998. Springer-Verlag.
- [SYY99] Tomas Sander, Adam Young, and Moti Yung. Non-Interactive CryptoComputing For NC<sup>1</sup>. In *40th Annual Symposium on Foundations of Computer Science*, 1999. IEEE Computer Society.
- [WY04] Rebecca N. Wright and Zhiqiang Yang. Privacy-Preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data. In *Proceedings of The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004. ACM.
- [ZD06] Paul Zimmermann and Bruce Dodson. 20 Years of ECM. In *The Algorithmic Number Theory Symposium*, volume 4076 of *Lecture Notes in Computer Science*, 2006. Springer-Verlag.
- [Zim06b] Paul Zimmermann. Optimal Parameters for ECM. Available at <http://www.loria.fr/~zimmerma/records/ecm/params.html>, as of May, 2006.

## A Non-interactive Partial Public Key Validation

Next, we propose another technique to transform the proposed protocols to be secure in the the standard model. It does not need extra messages but needs an extra amount of computations by an honest server. Namely, Thm. 3 holds even with incorrectly formed  $pk$  provided that the corresponding encryption rule is additively homomorphic and it is still possible to detect invalid ciphertexts. In particular, the Paillier cryptosystem is homomorphic even if a public modulus  $N$  is incorrectly formed. Thus, the verification of  $pk$  can just consist of computing a lower bound  $\gamma$  on factors of  $N$ . For small enough  $\gamma$  this can be done efficiently by using Lenstra's Elliptic Curve Method [Len87] which works in time  $\exp((\sqrt{2} + o(1))\sqrt{\ln p \cdot \ln \ln p})$  where  $p$  is the smallest factor of  $N$  [ZD06]. If we want the server's computation to be polynomial in  $\log N$  then we have to take a sufficiently small  $\ell$ . To provide some concrete numbers note that ECM allows "efficient" detection of 88-bit factors. Assume that the desired server-privacy level is  $2^{-40}$ . Such a choice of  $\varepsilon_2$  is most probably sufficient in practise. Then, in the case of the DIE protocol, one has  $\ell = 47$ , which is sufficient for several applications. In Spring 2006, we verified this approach by using the suggested optimal parameters from [Zim06b], on an AMD Athlon 64 3000+ processor by using the GMP-ECM software. As an example, if  $N = pq$ , where  $p$  is an 88-bit prime and  $q$  is an  $(1024 - 88)$ -bit prime then one has to run the ECM algorithm on an expected 206 curves with bounds  $B1 = 50\,000$  and  $B2 = 5\,000\,000$ . Testing on one curve with these parameters takes approximately 2.5 seconds, and thus testing that the smallest factor is greater than  $2^{89}$  takes 9 minutes on average. On the other hand, if  $q$  is an 66-bit prime then it takes an expected 77 curves with bounds  $B1 = 11\,000$  and  $B2 = 1\,100\,000$ . On the same platform, testing one curve with these parameters takes approximately 0.66 seconds and checking the bound  $2^{67}$  takes 51 seconds on average. Given the advances in the ECM, we would expect the quoted timings to decrease dramatically over the next few years.



### **Publication P3**

Ahto Buldas and Sven Laur. Knowledge-Binding Commitments with Applications in Time-Stamping. Lecture Notes in Computer Science, volume 4450, pages 150–165. © Springer-Verlag. Reprinted with permission.



# Knowledge-Binding Commitments with Applications in Time-Stamping

Ahto Buldas<sup>1,2,3,\*</sup> and Sven Laur<sup>4,\*\*</sup>

<sup>1</sup> Cybernetica AS, Akadeemia tee 21, 12618 Tallinn, Estonia.

<sup>2</sup> Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia.

<sup>3</sup> University of Tartu, Liivi 2, 50409 Tartu, Estonia. `Ahto.Buldas@ut.ee`

<sup>4</sup> Helsinki University of Technology, Laboratory for Theoretical Computer Science, P.O.Box 5400, FI-02015 TKK, Finland. `slaur@tcs.hut.fi`

**Abstract.** We prove in a non-black-box way that every bounded list and set commitment scheme is *knowledge-binding*. This is a new and rather strong security condition, which makes the security definitions for time-stamping much more natural compared to the previous definitions, which assume *unpredictability* of adversaries. As a direct consequence, list and set commitment schemes with partial opening property are sufficient for secure time-stamping if the number of elements has an explicit upper bound  $N$ . On the other hand, white-box reductions are in a sense strictly weaker than black-box reductions. Therefore, we also extend and generalize the previously known reductions. The corresponding new reductions are  $\Theta(\sqrt{N})$  times more efficient, which is important for global-scale time-stamping schemes where  $N$  is very large.

## 1 Introduction

Commitment schemes are basic building blocks in numerous cryptographic protocols. The most important properties of commitment schemes are binding and hiding. A commitment is hiding if it reveals no information about the committed message and binding if it is impossible to change the committed message afterwards without detection. First such schemes for committing a single bit were proposed by Blum [4] and by Brassard *et al* [5] and were proven secure under the hardness of factoring assumption. Later works have significantly improved their efficiency and weakened the underlying complexity theoretic assumptions, see [14, 10] for further references. Here, we study the so called *partially releasable* commitments, in which one can compute a commitment (also called *digest*) for a list  $\mathcal{X} = (x_1, \dots, x_N)$  of bit-strings, so that it is possible to partially open the commitment for every  $x_i \in \mathcal{X}$  without disclosing the other elements of  $\mathcal{X}$ . For opening  $x_i$  it is sufficient to present a decommitment string  $s_i$  (also called *certificate*). Achieving the hiding property is somewhat trivial, as one can always add another layer of commitments. Hence, our main emphasis is on the binding property. List commitments [3, 1, 17] that are only binding are known as *one-way accumulators*.

---

\* Partially supported by Estonian SF grant no. 6944, and by EU FP6-15964: “AEOLUS”.

\*\* Partially supported by Finnish Academy of Sciences, and by Estonian Doctoral School in Information and Communication Technologies.

In particular, we analyze the security of a *time-stamping* protocol, where clients send their requests  $x_1, \dots, x_N$  to a Time-Stamping Server (TSS) who computes the commitment  $c$  and sends the corresponding certificates  $s_1, \dots, s_N$  back to the clients. If  $c$  is published in an authentic way then everybody can verify that  $x_i$  was generated before  $c$  was published. This principle is used in practical time-stamping schemes [12] where  $c$  is computed as the root of a hash tree. List commitment schemes were believed to be exactly what one needs for such kind of time-stamping. However, Buldas *et al* [7] pointed out a flaw in the security proof of [12]. By giving a carefully crafted oracle separation they showed that pure collision-resistance is insufficient to prove that the hash tree time-stamping schemes [12] are secure. In other words, either there are collision-resistant functions that are still insecure for time-stamping, or the security of time-stamping schemes follows from currently unknown complexity-theoretic results. The key point of this paradoxical result is that the number of committed elements is potentially unbounded. In Sec. 4, we prove that all list and set commitments, where the cardinality of  $\mathcal{X}$  has an explicit bound  $|\mathcal{X}| \leq N$ , are suitable for time-stamping. The proof is given in the exact security framework and is  $\Theta(\sqrt{N})$  times more efficient than the previous reduction [7]. This improvement is especially valuable for global-scale time-stamping schemes in which  $N$  is very large.

In Sec. 5, we show that all binding bounded list and set commitments are *knowledge-binding*. This is a new and extremely strong security requirement inspired from the security of time-stamping schemes. Its strength is comparable to the *plaintext awareness* property, which is defined for public key encryption. The knowledge-binding property is also much more intuitive requirement for time-stamping schemes than the previous ones [7, 9], which use unpredictable probability distributions to model the stream of “new documents” sent to a TSS. Roughly, the knowledge-binding property states that for every efficient TSS, it is possible (by observing the commitment procedure) to efficiently extract the list  $\mathcal{X}$  of all documents that can be opened by the TSS in the future. The dedicated extractor must know only the internal coin tosses of TSS and some public parameters. Consequently, even if the TSS is malicious, it must *know* the whole list  $\mathcal{X}$  before the corresponding commitment is published. This allows to prove the security in the classical *ideal vs real world* comparison framework [11, pp.622–631,697–700].

Moreover, the notion of knowledge-binding commitments can be useful in other cryptographic protocols, because the ability to open a commitment does not change in time and we may skip the proofs of knowledge in the commitment phase. On the other hand, the corresponding security proofs are not black box. This means that once we have an efficient adversary  $A$  that breaks the knowledge-binding condition *we know that there exists* an efficient adversary  $A'$  that breaks the binding property of the corresponding commitment scheme. However, we may have no efficient ways to construct  $A'$ . Therefore, in reality the knowledge-binding property can be violated but the commitment scheme may still be practically binding—the efficient breaking procedure exists but is not known. Black-box security proofs in turn give an efficient procedure for constructing  $A'$  from  $A$ . In this sense, Theorems 1–4 give substantially stronger security guarantees for a fixed hash function (e.g. SHA-1) than Theorems 5 and 6.

In Sec. 6, we briefly discuss about other possible applications of knowledge-binding such as distributed and fine-grained time-stamping.

Some of the details of this work have been omitted because of space limitations. The missing details will be published in the IACR ePrint Archive.

## 2 Preliminaries and Notation

We use a non-uniform model of computations, where each algorithm  $A$  is specified as an input of a universal multi-tape Turing machine  $U$  that first copies the code of  $A$  to its working-tape and then starts to interpret it.  $A$  is a *t-time algorithm* if  $U$  performs at most  $t$  elementary operations to interpret the code of  $A$  independent of the input of  $A$ .

By  $x \leftarrow \mathcal{D}$  we mean that  $x$  is chosen randomly according to a distribution  $\mathcal{D}$ . In particular, if  $A$  is an algorithm, then  $x \leftarrow A(y)$  means that  $x$  is chosen according to the output distribution of  $A$  on an input  $y$ . Finite sets are identified with the corresponding uniform distributions, e.g.,  $x \leftarrow \{0, 1\}^\ell$  means that  $x$  is a uniformly chosen  $\ell$ -bit string. If  $\mathcal{D}_1, \dots, \mathcal{D}_m$  are distributions and  $F(x_1, \dots, x_m)$  is a predicate, then  $\Pr[x_1 \leftarrow \mathcal{D}_1, \dots, x_m \leftarrow \mathcal{D}_m : F(x_1, \dots, x_m)]$  denotes the probability that  $F(x_1, \dots, x_m)$  is true after the ordered assignment of  $x_1, \dots, x_m$ .

By a *cryptographic primitive*  $\mathfrak{P}$  we mean a set of computable functions associated with the advantage function  $\text{Adv}_{\mathfrak{P}}(\cdot)$ , such that for every adversarial algorithm  $A$ , the advantage  $\text{Adv}_{\mathfrak{P}}(A)$  is a positive real number. Mostly,  $\text{Adv}_{\mathfrak{P}}(A)$  is defined as the non-trivial success (scaled probability) in certain game  $\text{sec}$  that captures the desired properties of  $\mathfrak{P}$ . A primitive  $\mathfrak{P}$  is said to be  $(t, \varepsilon)$ -secure in terms of  $\text{sec}$  if  $\text{Adv}_{\mathfrak{P}}^{\text{sec}}(A) \leq \varepsilon$  for every  $t$ -time adversary  $A$ . For example, by a  $(t, \varepsilon)$ -secure *collision-resistant hash function* we mean a pair  $\mathcal{H} = (\text{Gen}, h)$  of algorithms such that if  $\text{pk} \leftarrow \text{Gen}$  is an arbitrary output of the generation function then  $h(\text{pk}, \cdot) = h_{\text{pk}}(\cdot)$  is a function of type  $\{0, 1\}^\ell \rightarrow \{0, 1\}^m$  where  $\ell > m$ ; and for every  $t$ -time adversary  $A$  :

$$\text{Adv}_{\mathcal{H}}^{\text{coll}}(A) = \Pr[\text{pk} \leftarrow \text{Gen}, (x_1, x_2) \leftarrow A(\text{pk}) : x_1 \neq x_2 \wedge h_{\text{pk}}(x_1) = h_{\text{pk}}(x_2)] \leq \varepsilon .$$

**Time-success ratio.** Quite often it is suitable for adversaries to find a trade-off between plausible attacking-time  $t$  and the corresponding advantage  $\varepsilon(t)$  against  $\mathfrak{P}$ . If the minimum *time-success ratio* for  $\mathfrak{P}$  is  $\alpha_{\mathfrak{P}}$ , then  $\varepsilon(t) \leq \frac{t}{\alpha_{\mathfrak{P}}}$  by definition. Often, we cannot estimate anything else about  $\mathfrak{P}$  than  $\alpha_{\mathfrak{P}}$ . Now, any black- or white-box reduction introduces a *change ratio*  $\gamma = \frac{\alpha_1}{\alpha_0}$  where  $\alpha_0$  is the time-success ratio of the basic primitive and  $\alpha_1$  is the ratio of the derived primitive, i.e., we have established a new approximate bound  $\varepsilon_1(t) \leq \frac{t}{\gamma \alpha_0}$ . Therefore, large values of  $\gamma$  provide better approximating bounds.

**Sampling bounds.** Our proofs use several standard statistical bounds. Let  $X_1, \dots, X_m$  be identically distributed independent zero-one random variables with  $\mu = \Pr[X_i = 1]$  and let  $X = \sum_{i=1}^m X_i$ . Then for any  $0 \leq \theta \leq 1$  the Chernoff bounds [13]

$$\Pr[X \leq (1 - \theta)\mu m] \leq e^{-\theta^2 m \mu / 2} , \quad \text{and} \quad \Pr[X \geq (1 + \theta)\mu m] \leq e^{-\theta^2 m \mu / 3} .$$

We also need a Birthday bound to determine the collision probability. Let  $Y_1, \dots, Y_m$  be identically but arbitrarily distributed independent random variables with possible values  $\{1, \dots, N\}$ . Then the probability  $p$  that all  $Y_i$ -s are different satisfies  $p \leq e^{-\frac{m(m-1)}{2N}}$ . In particular, if  $m \geq 1.5\sqrt{N}$  and  $N \geq 9$  then  $p \leq \frac{1}{2}$ .

### 3 Partially Releasable Commitment Schemes

**Set and list commitments.** Most commitment schemes for  $\ell$ -bit strings facilitate only complete disclosure of the committed input. In the context of time-stamping, the complete input can be several gigabytes long whereas we actually need to disclose only a few hundred bits. Therefore, we study commitment schemes that facilitate partial disclosure of inputs. *List commitments* are order-preserving: committed strings are ordered tuples. *Set commitments* in turn do not provide any ordering. Like ordinary commitment schemes, these commitments are specified by four basic algorithms: Gen, Com, Cert and Ver. Initialization algorithm Gen generates public parameters pk. Elements  $(m_1, \dots, m_n)$  are committed by computing  $(c, d) \leftarrow \text{Com}_{\text{pk}}(m_1, \dots, m_n)$ , where the commitment  $c$  is sent to the receiver and  $d$  is kept by the sender for later use. To prove that  $m_i$  was indeed used to compute the commitment  $c$ , the sender generates a certificate<sup>5</sup>  $s \leftarrow \text{Cert}_{\text{pk}}(d, m_i)$  the validity of which can be tested with the Ver algorithm.

The commitment scheme is *functional* if for any  $(c, d) \leftarrow \text{Com}_{\text{pk}}(m_1, \dots, m_n)$  and  $s \leftarrow \text{Cert}_{\text{pk}}(d, m_i)$ , the verification result  $\text{Ver}_{\text{pk}}(c, n, m_i, s) = \text{true}$  with overwhelming probability. For list commitments, the certificate  $s$  contains also the exact location  $i$  of the decommitted element, denoted as  $\text{loc}(s) = i$ . We explicitly assume that a decommitment certificate for a set  $\mathcal{X} = \{x_1, \dots, x_r\}$  is a union of the corresponding element certificates  $s_1, \dots, s_r$  denoted by  $s_1 \cup \dots \cup s_r$ . Consequently, certificates can be freely joined together and split into sub-certificates. For many commitment schemes such lists can further be compressed but this is only an implementation detail.

We omit the formal definition of the hiding property, since we study only the features related to the binding property. The binding property is different for set and list commitments. For list commitments, the binding property is violated if an adversary can open the  $i$ -th element in two different ways:

$$\text{Adv}^{\text{bind}}(\text{A}) = \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, (c, n, x_0, s_0, x_1, s_1) \leftarrow \text{A}(\text{pk}) : \\ x_0 \neq x_1 \wedge \text{loc}(s_0) = \text{loc}(s_1) \\ \wedge \text{Ver}_{\text{pk}}(c, n, x_0, s_0) = \text{Ver}_{\text{pk}}(c, n, x_1, s_1) = \text{true} \end{array} \right], \quad (1)$$

where the probability is taken over the coin tosses of all relevant algorithms. Since certificates are closed under union and there is no ordering for set commitments, the only way to misbehave is to exceed the size of  $\mathcal{X}$ :

$$\text{Adv}^{\text{bind}}(\text{A}) = \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, (c, n, \mathcal{X}, s) \leftarrow \text{A}(\text{pk}) : \\ \text{Ver}_{\text{pk}}(c, n, \mathcal{X}, s) = \text{true} \wedge |\mathcal{X}| > n \end{array} \right], \quad (2)$$

where  $\text{Ver}_{\text{pk}}(c, n, \mathcal{X}, s)$  first splits  $\mathcal{X}$  and  $s$  into components and then verifies each component  $x_i \in \mathcal{X}$  separately by using the corresponding component-certificate  $s_i \in s$ . We say that the commitment scheme is  $(\tau, \varepsilon)$ -binding if for all  $\tau$ -time adversaries  $\text{Adv}^{\text{bind}}(\text{A}) \leq \varepsilon$ . For unbounded adversaries, we speak about *statistical  $\varepsilon$ -binding*.

Note that set and list commitments must explicitly specify the number  $n$  of the committed elements. Indeed, if the certificates do not reveal the size of the commitment,

<sup>5</sup> To be precise, Cert should return a vector of certificates for each location of  $m_i$  in the list.

a malicious adversary can just hide some committed elements and receivers can never be sure if the commitment is fully opened. A commitment scheme is  $N$ -bounded if  $\text{Ver}_{\text{pk}}(c, n, x, s) = \text{false}$  for all  $n > N$ .

List commitment schemes that satisfy only the binding properties are known as *one-way accumulators* [1, 3, 17]. One-way accumulators that in addition to positive statements  $x \in \mathcal{X}$  also allow to (compactly) prove negative statements  $x \notin \mathcal{X}$  are called *undeniable attestors* [6]. The commonly used binding requirement for one-way accumulators is *n-times collision-freeness* [1], which is equivalent to the binding property of set commitments.

**Time-stamping schemes.** Time-stamping protocols process documents in batches  $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3, \dots$  that we call *rounds*. The rounds correspond to time periods of fixed duration (one hour, one day, etc.) After the  $i$ -th period, a short commitment  $c_i$  of the corresponding batch  $\mathcal{X}_i$  is published. A document  $x \in \mathcal{X}_i$  precedes document  $y$ , if there is  $j > 0$  such that  $y \in \mathcal{X}_{i+j}$ . Obviously, for a fixed commitment  $c_i$  there must be an efficient way to prove that  $x \in \mathcal{X}_i$ . However, for documents  $y \notin \mathcal{X}_i$  such proofs must be infeasible to create. Note that  $c_i$  can be viewed as a classical set or list commitment to the set  $\mathcal{X}_i$  and the corresponding proof of  $x \in \mathcal{X}_i$  as a certificate. Therefore, time-stamping schemes share the same functionality and algorithmic description as the set and list commitment schemes. Such a structural similarity is indeed remarkable. Still, careful studies of the security requirements reveal considerable differences between time-stamping and commitment schemes. Different security definitions exist for time-stamping schemes [7–9, 12]. In this paper, we adapt the strongest<sup>6</sup> definition [9] for the non-uniform precise security framework with minor modifications in notations.

Formal definitions of time-stamping schemes do not require that  $n$  is explicitly given as an argument to the verification algorithm  $\text{Ver}$ , but negative results in [7] suggest that time-stamping schemes (at least those without additional third parties) must be bounded, i.e.,  $n$  has to be at least implicitly specified.

Intuitively, time-stamping schemes must be secure against “back-dating” and this itself raises a subtle issue: How to model the future? Most works [7–9] have taken an approach based on computational entropy. Document generation is modeled as an efficient randomized procedure and the security guarantees are given for document distributions with high enough computational entropy. More formally, an adversary  $A = (A_1, A_2)$  is  $(\tau, \delta)$ -unpredictable if for every  $\tau$ -time predictor  $\Pi$  :

$$\text{Adv}_A^{\text{upr}}(\Pi) = \Pr \left[ \begin{array}{l} \omega_1 \leftarrow \Omega, \text{pk} \leftarrow \text{Gen}, \hat{x} \leftarrow \Pi(\text{pk}, \omega_1), \\ (c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1), (x, s) \leftarrow A_2(\phi) : \hat{x} = x \end{array} \right] \leq \delta ,$$

where  $\omega_1$  denotes the random coins of  $A_1$  and the probability is taken over the coin tosses of all relevant algorithms. The second stage  $A_2$  of the adversary models an efficient document generation (back-dating) procedure.

<sup>6</sup> There exist stronger security definitions for time-stamping schemes with additional (auditing) parties [8]. The main drawback of those schemes is a large amount of extra communication.

**Definition 1 (Entropy based security).** A time-stamping scheme is  $(t, \tau, \delta, \varepsilon)$ -secure if for every  $(\tau, \delta)$ -unpredictable  $t$ -time  $A$  :

$$\text{Adv}^{\text{ts}}(A) = \Pr \left[ \begin{array}{l} \omega_1 \leftarrow \Omega, \text{pk} \leftarrow \text{Gen}, (c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1), \\ (x, s) \leftarrow A_2(\phi) : \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \end{array} \right] \leq \varepsilon . \quad (3)$$

Here,  $\delta$  quantifies a trivial advantage. Indeed, consider the next adversary  $A = (A_1, A_2)$ :

- $A_1(\text{pk}; \omega_1)$  computes  $(c, d) \leftarrow \text{Com}_{\text{pk}}(\hat{x})$  and the corresponding valid certificate  $s \leftarrow \text{Cert}_{\text{pk}}(c, \hat{x})$  and outputs a tuple  $(c, 1, (\hat{x}, s))$ .
- $A_2(\hat{x}, s)$  generates a random  $x$  so that  $x = \hat{x}$  with probability  $\delta$ , and outputs  $(x, s)$ .

For every  $\tau$  the adversary  $A$  is  $(\tau, \delta)$ -unpredictable. However, no matter how the time-stamping scheme is defined, the advantage  $\text{Adv}^{\text{ts}}(A)$  of  $A$  is at least  $\delta$ . Hence, it is reasonable to assume that  $\delta \ll \varepsilon$ . Moreover, as  $\log \frac{1}{\delta}$  is an upper bound for the computational Rényi entropy, we implicitly assume that the computational Shannon entropy of the future documents is at least  $\log \frac{1}{\delta}$  w.r.t. the time-bound  $\tau$ .

The biggest drawback of the entropy based definition is non-uniformity. The security definition is natural in the polynomial model but has some flaws when adapted to the exact model. It only offers protection against  $(\tau, \delta)$ -unpredictable adversaries! Hence, it does not exclude extremely successful adversaries that are just *not quite so unpredictable*. In theory, a time-stamping scheme could be protected against  $(\tau, \delta)$ -unpredictable adversaries but still be totally insecure against  $(\tau, \delta + \delta^{100})$ -unpredictable adversaries. This flaw can be fixed by requiring strong uniformity in the definition:

**Definition 2 (Black-box security).** A time-stamping scheme is  $(t, \tau, \varepsilon)$ -secure if there exists a  $\tau$ -time black-box extractor machine  $\mathcal{K}$  such that for every  $t$ -time  $A$  :

$$\text{Adv}^{\text{ts}}(A) = \Pr \left[ \begin{array}{l} \omega_1 \leftarrow \Omega, \text{pk} \leftarrow \text{Gen}, \hat{\mathcal{X}} \leftarrow \mathcal{K}^{A(\text{pk}; \omega_1, \cdot)}(\text{pk}), \\ (c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1), (x, s) \leftarrow A_2(\phi) : \\ (\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \notin \hat{\mathcal{X}}) \vee |\hat{\mathcal{X}}| > n \end{array} \right] \leq \varepsilon , \quad (4)$$

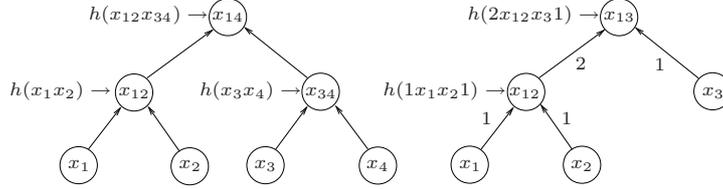
where  $\omega_1$  denotes random coins of  $A_1$  and  $\mathcal{K}$  gets a black-box access to  $A_1(\text{pk}; \omega_1)$  and  $A_2(\phi; \cdot)$ . The working time of  $\mathcal{K}^{A(\text{pk}; \omega_1, \cdot)}$  includes the time needed to execute all oracle calls. For list commitments, we treat  $\hat{\mathcal{X}}$  as a list and write  $x \in \hat{\mathcal{X}}$  iff  $x = \hat{\mathcal{X}}[\text{loc}(s)]$ .

Intuitively, we state that malicious time-stamping servers cannot issue valid certificates for unknown documents, as there exists a well known algorithm  $\mathcal{K}^{A(\text{pk}; \omega_1, \cdot)}$  for efficiently reconstructing the list of all valid documents  $\hat{\mathcal{X}}$ . This algorithm can be automatically constructed for every  $t$ -time adversary.

It is straightforward to see that  $(t, \tau, \varepsilon)$ -secure time-stamping scheme is always  $(t, \tau, \delta, \varepsilon + N\delta)$  secure where  $N \geq |\hat{\mathcal{X}}|$ , as one can use  $\mathcal{K}$  in prediction. In Sec. 4, we prove that every binding  $N$ -bounded list commitment scheme is also a secure time-stamping scheme. Still, there are quantitative differences between these two notions.

**Practical constructions based on hash trees.** Merkle trees [15] and count-certified hash trees [16] (described below) constructed from collision-resistant hash functions

are binding but not hiding even if the hash function is modeled as a random oracle—a release of an element (a leaf node) also reveals one neighboring element (the sibling leaf node). Nevertheless, if we use Merkle trees to compute a short commitment from hiding and binding commitments, we get binding and hiding list and set commitments.



**Fig. 1.** Merkle hash tree for  $\{x_1, x_2, x_3, x_4\}$  and a count-certified hash tree for  $\{x_1, x_2, x_3\}$ .

A *Merkle hash tree* for a list  $\mathcal{X}$  is a binary tree the leaves of which are the elements of  $\mathcal{X}$  and each non-leaf node is a hash of its two children (Fig. 1, left). Nodes with a single child can be avoided. Hence, every non-leaf node is assumed to have two children.

A *count-certified hash tree* (Fig. 1, right) is a binary tree which is similar to a Merkle tree, except that its arcs are labeled with *counters* each of which equal to the number of leaves in the corresponding subtree. Each non-leaf vertex  $v$  is a hash  $h(n_L x_L x_R n_R)$ , where  $n_L$  and  $n_R$  are the counters of the left- and the right subtree respectively. The counter  $c$  of the unique outgoing arc of  $v$  is the sum  $n_v = n_L + n_R$ .

Each hash tree can be represented as a commitment function  $(c, \mathcal{X}) \leftarrow \text{Com}_{\text{pk}}(\mathcal{X})$ , where  $c$  is the root hash value of the corresponding tree and  $\text{pk}$  denotes the public parameters associated with the collision-resistant hash function  $h$ . By the certificate  $\text{Cert}_{\text{pk}}(\mathcal{X}, x_i)$  for  $x_i \in \mathcal{X}$  we mean the smallest amount of data needed to recompute the root hash value. For example, in the Merkle hash tree (Fig. 1, left) the certificate  $s_2$  for  $x_2$  is  $s_2 = ((x_1, -), (-, x_{34}))$  which represents a sequence of hashing steps starting from the leaf  $x_2$  and ending with the root hash value, whereas  $-$  denotes an *empty slot* which during the verification is filled with the hash of the previous pair. Similarly, in the count-certified hash tree (Fig. 1, right) the certificate for  $x_2$  is  $s_2 = ((1, x_1, -, 1), (2, -, x_3, 1))$ . The verification function  $\text{Ver}_{\text{pk}}(c, n, x, s)$  simply recomputes the root hash value by using  $s$  and compares it with  $c$ . It also checks whether  $n \leq N$ . The verification algorithm for count-certified trees also recomputes the intermediate counter values to verify the certificate  $s$ , in particular if the counter of the root vertex is  $n$ .

**Collision-Extraction Property.** For hash trees with a fixed shape and count-certified hash trees there is a straight and precise reduction of the binding property to the collision-resistance of  $h$  because of the following property: If  $x_0 \neq x_1$ ,  $\text{Ver}_{\text{pk}}(c, n, x_0, s_0) = \text{Ver}_{\text{pk}}(c, n, x_1, s_1) = \text{true}$ , and  $\text{loc}(s_0) = \text{loc}(s_1)$ , then the internal  $h$ -calls of these two verifications comprise a collision for  $h$ . Moreover, if the tree is balanced, then the collision can be extracted in  $O(|s_0| + |s_1|) = O(\log_2 N)$  time.

1. Execute  $A_1$  in a black-box way and store  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$ .
2. Generate  $m$  independent samples  $(x_1, s_1) \leftarrow A_2(\phi), \dots, (x_m, s_m) \leftarrow A_2(\phi)$ .
3. Output  $(c, n)$  and a set of valid pairs  $\mathcal{V} = \{(x_i, s_i) : \text{Ver}_{\text{pk}}(c, n, x_i, s_i) = \text{true}\}$ .

**Fig. 2.** Black-box certificate extractor  $\mathcal{K}_{\text{cert}}^A(m)$ .

## 4 Bounded Commitments are Sufficient for Time-Stamping

In this section, we prove that bounded commitment schemes with partial opening are sufficient to construct secure time-stamping schemes. The new security reductions use a simple black-box certificate extractor (Fig. 2) and in the proofs we just show that a big enough set of valid decommitments  $\mathcal{V}$  allows to break the binding property.

Our proofs do not only generalize the existing ones [7] but are also more efficient. Presented theorems together with the previous separation results [7, 9] provide a clear border between the well studied classical binding properties like collision-freeness and the properties needed for time-stamping. For bounded commitment schemes the binding property implies time-stamping security. Otherwise, these notions are independent—binding properties are not necessary [9] nor sufficient [7].

To clarify the presentation, we have omitted a small  $O(N \log N + t)$  term that counts the computational effort needed to manage the list  $\mathcal{V}$  of valid decommitments, as the contribution to the total working time is irrelevant for all reasonable values of  $\varepsilon$ . To be absolutely precise, one has to increase the time-bounds for the binding property by  $O(N \log N + t)$  in Theorems 1–4.

**Theorem 1 (Entropy based security).** *Every  $(\frac{6t\sqrt{N}}{\varepsilon}, \frac{\varepsilon}{8})$ -binding and  $N$ -bounded list commitment scheme is also a  $(t, t, \frac{\varepsilon^3}{432 \cdot N}, \varepsilon)$ -secure time-stamping scheme for  $N \geq 9$ .*

*Proof.* Let  $A = (A_1, A_2)$  be a  $t$ -time adversary that violates  $(t, t, \frac{\varepsilon^3}{432 \cdot N}, \varepsilon)$ -security promise, i.e.,  $\text{Adv}^{\text{ts}}(A) \geq \varepsilon$  and  $A_2$  is sufficiently unpredictable (even for itself):

$$\Pr[\text{Coll}] := \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, (c, n, \phi) \leftarrow A_1(\text{pk}; \omega), \\ (x_0, s_0) \leftarrow A_2(\phi), (x_1, s_1) \leftarrow A_2(\phi) : x_0 = x_1 \end{array} \right] \leq \frac{\varepsilon^3}{432N}.$$

If  $m = \frac{6\sqrt{N}}{\varepsilon}$  then the black-box certificate extractor  $\mathcal{K}_{\text{cert}}^A(m)$  runs in time  $\frac{6t\sqrt{N}}{\varepsilon}$  and provides enough certificates to reveal a double opening. Let  $\text{Coll}^*$  denote that two equal messages  $x_i = x_j$  are produced internally by  $\mathcal{K}_{\text{cert}}^A(m)$ . Then by the union bound

$$\begin{aligned} \Pr[\text{Coll}^*] &\leq \sum_{\text{pk}, \omega_1} \Pr[\text{pk}, \omega_1] \cdot \frac{m(m-1)}{2} \cdot \Pr[\text{Coll} | \text{pk}, \omega_1] \\ &\leq \frac{m(m-1)}{2} \cdot \Pr[\text{Coll}] \leq \frac{m^2}{2} \cdot \frac{\varepsilon^3}{432N} \leq \frac{\varepsilon}{24}. \end{aligned}$$

Next, we estimate the number of valid document-certificate pairs created by  $\mathcal{K}_{\text{cert}}^A(m)$ . Let  $\varepsilon_{\text{pk}, \omega_1} = \text{Adv}^{\text{ts}}(A | \text{pk}, \omega_1)$  denote the probability that  $A$  is successful for fixed  $\text{pk}$

and  $\omega_1$ . As  $\Pr [\text{pk} \leftarrow \text{Gen}, \omega_1 \leftarrow \Omega : \varepsilon_{\text{pk}, \omega_1} \geq \frac{\varepsilon}{2}] \geq \frac{\varepsilon}{2}$ , we apply the Chernoff bound for these  $(\text{pk}, \omega_1)$  pairs with  $\theta = \frac{1}{2}$  and  $X_i$  indicating  $(x_i, s_i) \in \mathcal{V}$ , and get

$$\Pr [|\mathcal{V}| \leq 1.5\sqrt{N} | \varepsilon_{\text{pk}, \omega_1} \geq \frac{\varepsilon}{2}] \leq e^{-\frac{3\sqrt{N}}{8}} < 1/3 .$$

Since  $\mathcal{V}$  consists of identically distributed independent variables, we apply the Birthday bound. If  $|\mathcal{V}| \geq 1.5\sqrt{N}$  then  $\text{loc}(s_i) = \text{loc}(s_j)$  for some  $i, j$  with probability  $> \frac{1}{2}$ . Let  $C$  be an adversary that runs  $\mathcal{K}_{\text{cert}}^A(m)$  and then tries to find a double opening in  $\mathcal{V}$ . Then

$$\text{Adv}^{\text{bind}}(C) \geq \frac{\varepsilon}{2} \cdot \left(1 - e^{-\frac{3\sqrt{N}}{8}}\right) \cdot \frac{1}{2} - \Pr[\text{Coll}^*] > \frac{\varepsilon}{6} - \frac{\varepsilon}{24} = \frac{\varepsilon}{8}$$

for  $N \geq 9$  and we have obtained a desired contradiction.  $\square$

**Theorem 2 (Entropy based security).** *Every  $(\frac{4Nt}{\varepsilon}, \frac{\varepsilon}{8})$ -binding and  $N$ -bounded set commitment scheme is a  $(t, t, \frac{\varepsilon^3}{64N^2}, \varepsilon)$ -secure time-stamping scheme for  $N \geq 6$ .*

*Proof.* Similarly to the previous proof, let  $A = (A_1, A_2)$  be a  $t$ -time adversary that violates a  $(t, t, \frac{\varepsilon^3}{64N^2}, \varepsilon)$ -time-stamping security promise. In other words,  $\text{Adv}^{\text{ts}}(A) \geq \varepsilon$  and  $\Pr[\text{Coll}] \leq \frac{\varepsilon^3}{64(N+1)^2}$ . Fix  $m = \frac{4N}{\varepsilon}$ . Then the black-box certificate extractor  $C := \mathcal{K}_{\text{cert}}^A(m)$  then runs in time  $\frac{4Nt}{\varepsilon}$ . The Chernoff bound with  $\theta = \frac{1}{2}$  yields

$$\Pr [|\mathcal{V}| \leq N | \varepsilon_{\text{pk}, \omega_1} \geq \frac{\varepsilon}{2}] \leq e^{-\frac{N}{4}} < 1/2 .$$

Again,  $\Pr [\text{pk} \leftarrow \text{Gen}, \omega_1 \leftarrow \Omega : \varepsilon_{\text{pk}, \omega} \geq \frac{\varepsilon}{2}] \geq \frac{\varepsilon}{2}$  and we have obtained a contradiction:  $\text{Adv}^{\text{bind}}(C) \geq \frac{\varepsilon}{2} \cdot \left(1 - e^{-\frac{N}{4}}\right) - \Pr[\text{Coll}^*] > \frac{\varepsilon}{4} - \frac{m^2}{2} \cdot \frac{\varepsilon^3}{64N^2} = \frac{\varepsilon}{8}$ .  $\square$

**Theorem 3 (Uniform security).** *Every  $(\frac{2Nt}{\varepsilon}, \frac{\varepsilon}{2})$ -binding and  $N$ -bounded list commitment scheme is also  $(t, \frac{2Nt}{\varepsilon}, \varepsilon)$ -black-box secure time-stamping scheme.*

*Proof.* For the proof we have to fix a canonical black-box extractor machine  $\mathcal{K}^A$ :

1. First run  $A_1$  and store  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and set  $\hat{\mathcal{X}}[i] = \perp$  for  $i \in \{1, \dots, n\}$ .
2. Fix  $m = \frac{2N}{\varepsilon}$  and for  $k \in \{1, \dots, m\}$  do
  - Compute an independent sample  $(x_k, s_k) \leftarrow A_2(\phi)$ .
  - If  $\text{Ver}_{\text{pk}}(c, n, x_k, s_k) = \text{true}$  and  $\hat{\mathcal{X}}[\text{loc}(s_k)] = \perp$  then set  $\hat{\mathcal{X}}[\text{loc}(s_k)] = x_k$ .
3. Output the last snapshot of  $\hat{\mathcal{X}}$ .

Clearly, for every  $t$ -time adversary  $A = (A_1, A_2)$ , the extraction algorithm  $\mathcal{K}^A$  runs in time  $\frac{2Nt}{\varepsilon}$  and the extractor  $\mathcal{K}$  is valid for the definition.

For the sake of contradiction, assume that a  $t$ -time adversary  $A = (A_1, A_2)$  violates the security promise (4) w.r.t.  $\mathcal{K}$ . Let a pair  $(x_k, s_k)$  be *revealing* if  $x_k \neq \hat{\mathcal{X}}[\text{loc}(s_k)]$  in Step 2 of  $\mathcal{K}^A$ . Then the probability that  $(x_k, s_k)$  is revealing must be larger than  $\varepsilon$  for every  $k \in \{1, \dots, m\}$ , since the previous state of  $\hat{\mathcal{X}}$  can be viewed as a partial output of  $\mathcal{K}^A$ . Let  $X_k$  be the corresponding zero-one indicator variable, i.e.,  $X_k = 1$  if  $(x_k, s_k)$  is revealing. Then  $\varepsilon_k = \mathbf{E}[X_k] > \varepsilon$  and the average of  $S_m = \sum_{k=1}^m X_k$  is

$$\mathbf{E}[S_m] = \mathbf{E}[X_1 + \dots + X_m] = \varepsilon_1 + \dots + \varepsilon_m > m\varepsilon = 2N .$$

On the other hand,  $\mathbf{E}[S_m] \leq N + \Pr[S_m > N] \cdot \frac{2N}{\varepsilon}$  and thus  $\Pr[S_m > N] > \frac{\varepsilon}{2}$ . Therefore, with probability strictly more than  $\frac{\varepsilon}{2}$  there are  $N + 1$  revealing pairs  $(x_k, s_k)$  computed by  $\mathcal{K}^A$ . As the commitment scheme is  $N$ -bounded, revealing pairs exist only if  $n \leq N$ . Hence, at least one slot must be overwritten if there are  $N + 1$  revealing pairs and we have found a double opening with probability strictly more than  $\frac{\varepsilon}{2}$ .  $\square$

**Theorem 4 (Uniform security guarantee).** *Every  $(\frac{2Nt}{\varepsilon}, \frac{\varepsilon}{2})$ -binding  $N$ -bounded set commitment scheme is also  $(t, \frac{2Nt}{\varepsilon}, \varepsilon)$ -black-box secure time-stamping scheme.*

*Proof.* The construction given above is also valid for set commitments.  $\square$

**Comparison with previous results.** Our reductions are not completely novel. A similar proof with a different reduction was given in [7] for hash trees. Therefore, we compare the time-success ratios. Recall that the minimal time-success ratio  $\alpha$  implies  $\varepsilon(t) \leq \frac{t}{\alpha}$  and hence large ratios  $\gamma = \frac{\alpha_1}{\alpha_0}$  lead to better security bounds.

In Thm. 1 we constructed a double opener with running time  $t_0 \approx \frac{6t\sqrt{N}}{\varepsilon}$  and with advantage  $\varepsilon_0 \approx \frac{\varepsilon}{8}$ , based on a back-dating adversary with running time  $t$  and advantage  $\varepsilon$ . Thus the change ratio is  $\gamma \approx \frac{\varepsilon}{48\sqrt{N}}$  for our reduction. If we adapt the reduction presented in [7] for the exact security model we obtain a ratio  $\gamma \approx \frac{\varepsilon}{2N}$ , which is significantly smaller for  $N \geq 600$ . In *global-scale* time-stamping services,  $N$  can be very large (say millions or even billions) and our new reduction by far supersedes the previous one [7].

Similarly, one can verify that  $\gamma \approx \frac{\varepsilon}{4N}$  for Thm. 3 and Thm. 4 but the security guarantees are much stronger. To break the black-box security an adversary can produce valid document-certificate pairs with low computational Rényi entropy, which makes it impossible to use the birthday paradox. It is easy to see that the extractor must work in time  $\Theta(\frac{Nt}{\varepsilon})$  and  $\sqrt{N}$  in the denominator is not achievable.

## 5 All Bounded Commitment Schemes are Knowledge-Binding

Both security definitions for time-stamping (Def. 1,2) are based on heuristic assumptions. Namely, the future is modeled as a *computationally efficient stochastic process*. Such an assumption has two major drawbacks. Firstly, it is philosophically questionable and causes practical problems in the classical framework of secure computations [11]: due to the non-uniform nature of such model, future documents may have arbitrary distributions. Secondly, the success of back-dating adversaries is computed as an average over the distribution of future documents and it might still be easy to “backdate” a fixed document. To overcome these problems, we propose a new security notion where the future is modeled as an advice string that is independent of pk. The independence assumption is essential. Otherwise, no computationally binding commitment scheme can be secure, since the advice may contain explicit double-openings.

**Definition 3.** A commitment scheme is  $(t, \tau, \varepsilon)$ -knowledge-binding if for every  $t$ -time adversary  $A = (A_1, A_2)$  there exist a dedicated  $\tau$ -time extractor machine  $\mathcal{K}_A$  such that

$$\text{Adv}^{\text{k-bind}}(A) = \max_{\text{adv}} \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, \omega_1 \leftarrow \Omega, \hat{\mathcal{X}} \leftarrow \mathcal{K}_A(\text{pk}; \omega_1), \\ (c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1), (x, s) \leftarrow A_2(\phi, \text{adv}) : \\ (\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \notin \hat{\mathcal{X}}) \vee |\hat{\mathcal{X}}| > n \end{array} \right] \leq \varepsilon ,$$

where  $\text{adv}$  varies over all advices of length  $t$  and the probability is taken over the coins of  $\text{Gen}$ ,  $A_1$  and  $A_2$ . For list commitments,  $\hat{\mathcal{X}}$  is a list and write  $x \in \hat{\mathcal{X}}$  iff  $x = \hat{\mathcal{X}}[\text{loc}(s)]$ .

The new definition explicitly states that there exists an efficient *extraction strategy*  $\mathcal{K}_A$  that is able (by observing the internal computations of the committing algorithm  $A_1$ ) to predict any bit-string  $x$  that is later "back-dated" by  $A_2$ . I.e, in some sense  $x$  already existed before the commitment and no real back-dating attacks were performed.

But there is an even more intuitive interpretation. When an adversary publishes a commitment  $c$ , he implicitly fixes his level of knowledge about the commitment and no future actions can change it. As the level of knowledge does not change in time, a successful opening "proves" that the adversary already "knew" the committed element when the commitment was created. Hence, we can omit proofs of knowledge at the commitment stage and reduce the number of rounds in various protocols. Thus, the new notion is very similar to plaintext-awareness of public-key encryption schemes.

Finally, note that knowledge-binding is a necessary condition for the multi-party security of time-stamping schemes. In the ideal implementation, TSS gives a list  $\mathcal{X}$  to a trusted party who will later serve partial release queries  $x \in \mathcal{X}$ ? Hence, there must be an efficient way to extract all documents that TSS can potentially open as a response for any future message that is independent of  $\text{pk}$ , i.e., the extractor machine  $\mathcal{K}_A$  must exist. To get multi-party security in the malicious model, we must also protect a honest TSS against malicious clients. This can be done in an obvious way by using digital signatures, but due to the space limitations we defer the discussion to follow-up articles.

Clearly, the knowledge-binding property can be established only by using white-box reductions. In other words, we cannot efficiently construct the code of  $\mathcal{K}_A$  given only the code of  $A$ , although  $\mathcal{K}_A$  itself is an efficient algorithm. Such reductions provide substantially weaker security guarantees for *fixed hash functions* like SHA-1, since we know *a priori* that efficient collision finders must exist for SHA-1. Therefore, the claims of existence without efficient construction strategies provide no new information. As a result, we can only talk about the security of hash function families, i.e., we have to consider SHA-1 as a "typical" representative of a collision-free hash function family.

The proofs consist of two main steps. First we analyze the behavior of  $A$  and construct a dedicated knowledge extractor  $\mathcal{K}_A$ . Next we show that  $\mathcal{K}_A$  is efficient and  $\text{Adv}^{\text{k-bind}}(A)$  is sufficiently small. To construct  $\mathcal{K}_A$ , we run  $A$  on all possible inputs and find suitable triggering messages  $\text{adv}$  that force  $A$  to reveal most of the valid certificates. Next, we construct  $\mathcal{K}_A$  from  $A$  and the triggering messages. As the knowledge-binding condition only requires the *existence* of  $\mathcal{K}_A$ , the construction time is not an issue.

**Theorem 5.** For every  $t > 0$  and  $\delta > 0$ , there exists  $\tau = (\frac{N}{\delta} + 1) \cdot O(t)$  such that every  $(\tau, \varepsilon)$ -binding list commitment scheme is  $(t, \tau, \varepsilon + \delta)$ -knowledge binding.

*Proof.* Fix a  $t$ -time adversary  $A$  and consider a giant status matrix  $W[\text{pk}, \omega_1; \text{adv}, \omega_2]$  the rows of which are indexed by public keys  $\text{pk}$  and random coins  $\omega_1$  of  $A_1$ , whereas the columns are indexed by  $t$ -bit advices  $\text{adv}$  and random coins  $\omega_2$  of  $A_2$ . Define

$$W[\text{pk}, \omega_1; \text{adv}, \omega_2] = \begin{cases} 0, & \text{if } \text{Ver}_{\text{pk}}(c, n, x, s) = \text{false} , \\ \text{loc}(s), & \text{if } \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} , \end{cases}$$

where  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and  $(x, s) \leftarrow A_2(\phi, \text{adv}; \omega_2)$ . Note that few columns of  $W$  cover most of the rows containing non-zero elements. Namely, Lemma 1 from App. A assures the existence of  $\mathcal{I} = \{(\text{adv}_1, \omega_2^1), \dots, (\text{adv}_k, \omega_2^k)\}$  such that  $|\mathcal{I}| \leq \frac{N}{\delta}$  and for any fixed advice-randomness pair  $(\text{adv}, \omega_2)$ :

$$\Pr [(\text{pk}, \omega_1) : 0 \neq W[\text{pk}, \omega_1; \text{adv}, \omega_2] \notin \mathcal{L}[\text{pk}, \omega_1] \wedge |\mathcal{L}[\text{pk}, \omega_1]| < N] \leq \delta , \quad (5)$$

where  $\mathcal{L}[\text{pk}, \omega_1] = \{W[\text{pk}, \omega_1; \text{adv}, \omega_2] : (\text{adv}, \omega_2) \in \mathcal{I}\}$  is a set of revealed locations. Now the construction<sup>7</sup> of  $\mathcal{K}_A$  is evident:

1. Given  $(\text{pk}, \omega_1)$  store  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and set  $\hat{X}[i] = \perp$  for  $i \in \{1, \dots, n\}$ .
2. For each  $(\text{adv}, \omega_2) \in \mathcal{I}$  do
  - Compute  $(x, s) \leftarrow A_2(\phi, \text{adv}; \omega_2)$ .
  - If  $\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true}$  then set  $\hat{X}[\text{loc}(s)] \leftarrow x$ .
3. Output the last snapshot of  $\hat{X}$ .

To analyze the advantage of  $\mathcal{K}_A$ , we fix a pair  $(\text{adv}, \omega_2)$ . Let  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and  $(x, s) \leftarrow A_2(\phi, \text{adv}; \omega_2)$  as before. For valid decommitment value  $s$ , the entry  $\hat{X}[\text{loc}(s)] = \perp$  only if  $|\mathcal{L}[\text{pk}, \omega_1]| < N$  and thus the inequality (5) given above yields  $\Pr [(\text{pk}, \omega_1) : \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge \hat{X}[\text{loc}(s)] = \perp] \leq \delta$ . Alternatively,  $\mathcal{K}_A$  can fail if  $\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true}$  but  $\hat{X}[\text{loc}(s)] \neq x$ . However, we can naturally combine  $A_1$ ,  $A_2$  and  $\mathcal{K}_A$  into an adversary  $B$  that outputs these double openings and performs  $(\frac{N}{\delta} + 1) \cdot O(t)$  elementary operations. Consequently,  $\text{Adv}^{\text{bind}}(B) \leq \varepsilon$  and thus

$$\Pr [(\text{pk}, \omega_1) : \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \neq \hat{X}[\text{loc}(s)] \neq \perp] \leq \varepsilon .$$

As a result, we have obtained that for any pair  $(\text{adv}, \omega_2)$ :

$$\Pr [(\text{pk}, \omega_1) : \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \neq \hat{X}[\text{loc}(s)]] \leq \delta + \varepsilon$$

and the claim follows.  $\square$

**Theorem 6.** *For every  $t > 0$  and  $\delta > 0$ , there exists  $\tau = (\frac{N}{\delta} + 1) \cdot O(t)$  such that every  $(\tau, \varepsilon)$ -binding set commitment scheme is  $(t, \tau, \varepsilon + \delta)$ -knowledge-binding.*

<sup>7</sup> Note that all elements of the set  $\mathcal{I}$  are hardwired as explicit constants into the code of  $\mathcal{K}_A$ , i.e.,  $\mathcal{K}_A$  does not compute  $\mathcal{I}$ . As  $\mathcal{K}_A$  runs on a universal Turing machine, it must rewind the code of  $A_2$  and thus  $\mathcal{K}_A$  performs at most  $O(t)$  extra steps to complete the loop of Step 2.

*Proof.* Fix a  $t$ -time adversary  $A$  and consider a status matrix  $W[\text{pk}, \omega_1; \text{adv}, \omega_2]$  that is indexed identically to the previous proof but the entries are defined differently:

$$W[\text{pk}, \omega_1; \text{adv}, \omega_2] = \begin{cases} 0, & \text{if } \text{Ver}_{\text{pk}}(c, n, x, s) = \text{false} , \\ x, & \text{if } \text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} , \end{cases}$$

where  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and  $(x, s) \leftarrow A_2(\phi, \text{adv}; \omega_2)$ . Then Lemma 1 from App. A assures the existence of  $\mathcal{I} = \{(\text{adv}_1, \omega_2^1), \dots, (\text{adv}_k, \omega_2^k)\}$  such that  $|\mathcal{I}| \leq \frac{N}{\delta}$  and for every fixed advice-randomness pair  $(\text{adv}, \omega_2)$ :

$$\Pr [(\text{pk}, \omega_1) : 0 \neq W[\text{pk}, \omega_1; \text{adv}, \omega_2] \notin \mathcal{L}[\text{pk}, \omega_1] \wedge |\mathcal{L}[\text{pk}, \omega_1]| < N] \leq \delta , \quad (6)$$

where  $\mathcal{L}[\text{pk}, \omega_1] = \{W[\text{pk}, \omega_1; \text{adv}, \omega_2] : (\text{adv}, \omega_2) \in \mathcal{I}\}$  is a set of revealed elements. Now the construction of  $\mathcal{K}_A$  is straightforward:

1. Given  $(\text{pk}, \omega_1)$  store  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and set  $\hat{\mathcal{X}} \leftarrow \emptyset$ .
2. For each  $(\text{adv}, \omega_2) \in \mathcal{I}$  do
  - Compute  $(x, s) \leftarrow A_2(\phi, \text{adv}; \omega_2)$ .
  - If  $\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true}$  then add  $x$  to  $\hat{\mathcal{X}}$ .
3. Output the last snapshot of  $\hat{\mathcal{X}}$ .

To analyze the advantage of  $\mathcal{K}_A$ , fix  $(\text{adv}, \omega_2)$ . Let  $(c, n, \phi) \leftarrow A_1(\text{pk}; \omega_1)$  and  $(x, s) \leftarrow A_2(\phi, \text{adv}, \omega_2)$  as before. As  $\hat{\mathcal{X}}[\text{pk}, \omega_1] = \mathcal{L}[\text{pk}, \omega_1]$  by the construction (see Lemma 1), the inequality (6) yields  $\Pr [\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \notin \hat{\mathcal{X}} \wedge |\hat{\mathcal{X}}| < n \leq N] \leq \delta$ . The extractor  $\mathcal{K}_A$  can also fail when  $\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true}$  but  $x \notin \hat{\mathcal{X}}$  and  $|\hat{\mathcal{X}}| \geq n$ . Again, we can naturally combine  $A_1, A_2$  and  $\mathcal{K}_A$  into an adversary  $B$  with running-time  $(\frac{N}{\delta} + 1) \cdot O(t)$  that runs all algorithms and extracts all valid openings. Consequently, the restriction  $\text{Adv}^{\text{bind}}(B) \leq \varepsilon$  yields  $\Pr [\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \notin \hat{\mathcal{X}} \wedge |\hat{\mathcal{X}}| \geq n] \leq \varepsilon$  and we have obtained that for any pair  $(\text{adv}, \omega_2)$ :

$$\Pr [\text{Ver}_{\text{pk}}(c, n, x, s) = \text{true} \wedge x \notin \hat{\mathcal{X}}] \leq \delta + \varepsilon$$

and the claim follows.  $\square$

**Efficiency of the new reduction.** Again, we compute time-success ratios to compare the efficiency of the new white-box reduction to the previous black-box ones. To have a fair comparison we take  $\delta \approx \varepsilon$ . Then Theorems 5 and 6 provide attacks against the binding property with parameters  $t_0 \approx (\frac{N}{\delta} + 1)t$  and  $\varepsilon_0 = \varepsilon$ , provided that there exist a  $t$ -time adversary achieving  $\varepsilon + \delta$  success. As a result, we obtain a change ratio  $\gamma = \frac{\alpha_1}{\alpha_0} \approx (\frac{N}{\delta} + 1)^{-1} \cdot \frac{\varepsilon}{\varepsilon + \delta} \approx \frac{\varepsilon}{2N}$ , which is better than the change ratio  $\gamma \approx \frac{\varepsilon}{4N}$  provided by Thm. 3 and Thm. 4. The difference is not essential rather it comes from slightly loose success bounds in Thm. 3 and Thm. 4.

## 6 Applications of Knowledge-Binding Commitments

Here, we briefly describe how knowledge-binding count-certified hash trees can be used and why knowledge-binding property is important. Knowledge-binding property can be

viewed as an indifference against outside advices. Similar to the plaintext-awareness, the knowledge-binding property allows one to combine commitments with other cryptographic primitives without a fear of unwanted interference. Such interference often makes it hard or impossible to prove the security of new constructions. If the secret or public parameters of other primitives are independent of the commitment parameters  $pk$ , then the rest of the protocol can be interpreted as an external advice. Hence, one can use the standard hybrid argument technique even if the primitives are used concurrently.

**Distributed and fine-grain time-stamping.** Knowledge-binding commitments give rise to a secure time-stamping service where a central time-stamping authority (TSS) computes and publishes the round commitment  $(c, n)$  and distributes the respective certificates  $s_i$  to the clients. But such service is susceptible to denial-of-service attacks. Hence, it is more natural to consider a distributed service where  $k$  independent servers compute sub-commitments  $(c_i, n_i)$  and at the end of the round the master commitment  $(c, n)$  is compiled. Therefore, it is advantageous to use knowledge-binding commitments that facilitate fast merging of sub-commitments and mostly local certificate computations. Count-certified hash trees have the following important property: every root node  $(c_i, n_i)$  of a hash subtree forms a correct commitment. Moreover, given two root nodes  $(c_L, n_L)$  and  $(c_R, n_R)$  it is straightforward to compute the commitment of the merged tree and update the corresponding certificates.

In a way, a set commitment scheme provides a really coarse-grain time-stamping service. It is impossible to order the events inside the round  $\mathcal{X}$ . List commitment provides only a partial solution, as clients have to trust that the TSS orders documents correctly in a single round. Tree-shaped list commitments that preserve knowledge-binding w.r.t. the root of each subtree allow also fine-grained time-stamping even if the TSS acts maliciously. Essentially, TSS has to send to a Client all root commitments  $(c_i, n_i)$  of all preceding computations, then the Client has strong guarantees that after submitting his query the TSS cannot insert any messages in the prefix of the list without getting caught. Hence, count-certified hash trees could be used for fine-grain time-stamping.

**Non-malleable partially releasable commitments.** To show that knowledge-binding commitments have other applications outside of the domain of time-stamping, we give a construction of partially releasable non-malleable commitments from non-malleable string commitments and knowledge-binding commitments. It is just an informal example, we do not formalize the claim due to the lack of space.

Recall that a commitment scheme is non-malleable if given a commitment  $c$  it is infeasible to construct a new commitment  $c' \neq c$  such that after seeing a certificate  $s$  for  $x$  it is infeasible to output a valid certificate  $s'$  for  $x'$  such that  $x$  and  $x'$  are related. Let  $\mathcal{L} = \{c_1, \dots, c_n\}$  be a list of non-malleable commitments for  $x_1, \dots, x_n$  and  $(C, D) \leftarrow \text{Com}_{pk}(\mathcal{L})$  is computed by using a knowledge-binding commitment scheme. Then the resulting commitment scheme is non-malleable. From the knowledge-binding property it follows that after seeing a proof that  $c_i$  was computed by using  $x_i$ , adversary's ability to output certificates  $(c, s)$  such that  $\Pr[\text{Ver}(C, n, c, s) = \text{true}]$  does not increase. Hence, the adversary knows all valid commitment-certificate pairs  $(c_i, s_i)$  essentially before any commitment is opened. Therefore, non-malleability directly follows from the non-malleability of the lower-level commitment.

## References

1. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Proc. of *EUROCRYPT'97, LNCS 1233*, pages 480–494, 1997.
2. D. Bayer, S. Haber, and W.-S. Stornetta. Improving the efficiency and reliability of digital time-stamping. In *Sequences II: Methods in Communication, Security, and Computer Science*, pages 329–334, Springer-Verlag, New York 1993.
3. J. Benaloh and M. de Mare. One-way accumulators: a decentralized alternative to digital signatures. In Proc. of *EUROCRYPT'93, LNCS 765*, pages 274–285, 1994.
4. M. Blum. Coin flipping by telephone: a protocol for solving impossible problems. In Proc. of *CompCon*, pages 133–137, 1982.
5. G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *JCSS*, vol.37, pages 156–189, 1988.
6. A. Buldas, P. Laud, H. Lipmaa. Eliminating counterevidence with applications to accountable certificate management. *Journal of Computer Security*, 10(3), pages 273–296, 2002.
7. A. Buldas and M. Saarepera. On provably secure time-stamping schemes. In Proc. of *ASIACRYPT 2004, LNCS 3329*, pages 500–514, 2004.
8. A. Buldas, P. Laud, M. Saarepera, and J. Willemsen. Universally composable time-stamping schemes with audit. In *ISC05, LNCS 3650*, pages 359–373, 2005.
9. A. Buldas, S. Laur. Do broken hash functions affect the security of time-stamping schemes? In Proc. of *ACNS'06, LNCS 3989*, pages 50–65, 2006.
10. I. Damgård. Commitment schemes and zero knowledge protocols. In *Lectures on Data Security: modern cryptology in theory and practice, LNCS 1561*, pages 63–86, 1999.
11. O. Goldreich. *Foundations of Cryptography II: Basic Applications*, Cambridge University Press, 2004.
12. S. Haber and W.-S. Stornetta. Secure Names for Bit-Strings. In Proc. of *ACM Conference on Computer and Communications Security*, pages 28–35, 1997.
13. T. Hagerup and C. Rüb. A Guided Tour of Chernoff Bounds. *Information Processing Letters*, 33, pages 305–308, 1990.
14. S. Halevi and S. Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *CRYPTO'96, LNCS 1109*, pages 201–215, 1996.
15. R. C. Merkle. Protocols for public-key cryptosystems. *Proceedings of the 1980 IEEE Symposium on Security and Privacy*, pages 122–134, 1980.
16. G. Nuckolls, C. U. Martel, and S. G. Stubblebine. Certifying Data from Multiple Sources. In Proc. of the *DBSec 2003*, pages 47–60, 2003.
17. K. Nyberg. Fast accumulated hashing. In Proc. of *FSE'96, LNCS 1039*, pages 83–87, 1996.

## A Combinatorial Extraction Lemma

Consider a finite matrix  $W[r; c]$  the rows of which are indexed by  $r \in \mathcal{R}$  and the columns are indexed by  $c \in \mathcal{C}$ . Moreover, assume that a certain probability measure  $\Pr[\cdot]$  is defined over the row indices  $\mathcal{R}$ . Then it is straightforward to state and prove a combinatorial lemma that we used for proving the knowledge-binding property.

**Lemma 1.** *For any  $\delta > 0$  and  $N \in \mathbb{N}$ , there exist a set of column indices  $\emptyset \subseteq \mathcal{I} \subseteq \mathcal{C}$  such that  $0 \leq |\mathcal{I}| \leq \frac{N}{\delta}$  and for every column  $c \in \mathcal{C}$ :*

$$\Pr[r \leftarrow \mathcal{R} : W[r; c] \neq 0 \wedge W[r; c] \notin \mathcal{L}[r] \wedge |\mathcal{L}[r]| < N] \leq \delta ,$$

where  $\mathcal{L}[r] = \{W[r; c] : c \in \mathcal{I}\} \setminus \{0\}$  is the set of nonzero elements revealed by  $\mathcal{I}$ .

*Proof.* Consider following iterative procedure:

1. Set  $\mathcal{I} = \emptyset$  and initialise row counters  $\text{cnt}[r] = N$  for  $r \in \mathcal{R}$ .
2. While exists  $c \in \mathcal{C}$  such that  $\Pr[r : W[r; c] \neq 0] \geq \delta$  do
  - (a) Choose  $c$  such that  $\Pr[r : W[r; c] \neq 0] \geq \delta$  and insert  $c$  into  $\mathcal{I}$ .
  - (b) For each row  $r \in \mathcal{R}$  such that  $W[r; c] \neq 0$  do
    - Store  $w \leftarrow W[r; c]$ .
    - Remove  $w$  entries from the row.
    - If  $W[r; c'] = w$  then  $W[r; c'] \leftarrow 0$  for  $c' \in \mathcal{C}$ .
    - Decrease counter  $\text{cnt}[r] \leftarrow \text{cnt}[r] - 1$ .
  - (c) Zero all rows where  $\text{cnt}[r] = 0$ .
    - If  $\text{cnt}[r] = 0$ , set  $W[r; c'] \leftarrow 0$  for  $c' \in \mathcal{C}$ .

Let  $\mathcal{N} = \{r : \exists W[r; c] \neq 0\}$  denote nonzero rows and  $\mathcal{N}_{\text{old}}, \mathcal{N}_{\text{new}}$  denote the value of  $\mathcal{N}$  before and after update at Step 2. Let

$$\mu[\mathcal{N}] = \sum_{r \in \mathcal{N}} \Pr[r] \text{cnt}[r]$$

be the average counter value. Then by the construction  $\mu[\mathcal{N}_{\text{new}}] \leq \mu[\mathcal{N}_{\text{old}}] - \delta$  after a single iteration of Step 2. As initially  $\mu[\mathcal{N}] \leq N$ , then after  $\lfloor N/\delta \rfloor$  iterations  $\Pr[\mathcal{N} \leq \mu[\mathcal{N}] < \delta] < \delta$ . Note that the algorithm nullifies the elements  $W[r; c]$  only if they already belong to  $\mathcal{L}[r]$  or  $|\mathcal{L}[r]| \geq N$ . In the end, each column  $c$  contains at most a  $\delta$ -fraction of elements that satisfy the predicate  $W[r; c] \neq 0 \wedge W[r; c] \notin \mathcal{L}[r] \wedge |\mathcal{L}[r]| < N$  and the claim follows. Note that  $\mathcal{I}$  can be empty.  $\square$

| $\mathcal{I} = \emptyset$                                                                                                             | $\mathcal{L}$ | $\mathcal{I} = \{1\}$                                                                                                                                                                                                                                                                                                                                                  | $\mathcal{L}$ | $\mathcal{I} = \{1, 3\}$                                                                                                                                                                                                                                                                                                                                               | $\mathcal{L}$ | $\mathcal{I} = \{1, 3\}$                                                                                                                                                                                                                                                                                                                                               | $\mathcal{L}$ |             |
|---------------------------------------------------------------------------------------------------------------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|-------------|
| $\begin{bmatrix} 1 & 2 & 0 & 1 & 1 \\ 1 & 0 & 3 & 0 & 2 \\ 2 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$ | $\emptyset$   | $\begin{bmatrix} \mathbf{0} & \mathbf{2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{3} & \mathbf{0} & \mathbf{2} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{2} \end{bmatrix}$ | $\{1\}$       | $\begin{bmatrix} \mathbf{0} & \mathbf{2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{2} \end{bmatrix}$ | $\{1, 3\}$    | $\begin{bmatrix} \mathbf{1} & \mathbf{2} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{0} & \mathbf{3} & \mathbf{0} & \mathbf{2} \\ \mathbf{2} & \mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{2} \end{bmatrix}$ | $\{1, 3\}$    | $\emptyset$ |
|                                                                                                                                       | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        |               |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |             |
|                                                                                                                                       | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        |               |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |             |
|                                                                                                                                       | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |                                                                                                                                                                                                                                                                                                                                                                        |               |                                                                                                                                                                                                                                                                                                                                                                        | $\Rightarrow$ |             |

**Fig. 3.** Illustration of Lemma 1. The first three sub-figures show how the columns are selected for the uniform distribution over the rows and for parameter values  $N = 2$ ,  $\delta = 0.3$ , boldface symbols denote the changed values. The last sub-figure shows the final result. Boldface symbols denote the revealed entries. Underlined symbols denote the entries that satisfy the predicate.

## **Publication P4**

Sven Laur and Kaisa Nyberg. Efficient Mutual Data Authentication Using Manually Authenticated Strings. Lecture Notes in Computer Science, volume 4301, pages 90–107. © Springer-Verlag. Reprinted with permission.



# Efficient Mutual Data Authentication Using Manually Authenticated Strings

Sven Laur<sup>2</sup> and Kaisa Nyberg<sup>1,2</sup>

<sup>1</sup> Nokia Research Center, Finland [kaisa.nyberg@nokia.com](mailto:kaisa.nyberg@nokia.com)

<sup>2</sup> Helsinki University of Technology, Finland [{slaur,knyberg}@tcs.hut.fi](mailto:{slaur,knyberg}@tcs.hut.fi)

**Abstract.** Solutions for an easy and secure setup of a wireless connection between two devices are urgently needed for WLAN, Wireless USB, Bluetooth and similar standards for short range wireless communication. All such key exchange protocols employ data authentication as an unavoidable subtask. As a solution, we propose an asymptotically optimal protocol family for data authentication that uses short manually authenticated out-of-band messages. Compared to previous articles by Vaudenay and Pasini the results of this paper are more general and based on weaker security assumptions. In addition to providing security proofs for our protocols, we focus also on implementation details and propose practically secure and efficient sub-primitives for applications.

## 1 Introduction

In this paper we consider the problem of setting up a shared secret key in an ad hoc manner, that is, in isolation from any key management facility and without pre-shared secrets. Consider two parties Alice and Bob who want to establish a shared secret key over an insecure network without any prior authenticated information. If adversaries are passive, that is, no malicious messages are sent to the network and all messages are delivered unaltered, then exchanging public keys for Diffie-Hellman or similar public key based key exchange protocol is sufficient. However, an active adversary, Charlie, can launch a man-in-the-middle attack. Namely, Charlie can replace a desired secure channel from Alice to Bob by a pair of secure channels, one from Alice to Charlie and one from Charlie to Bob. The attack is transparent to legitimate users without prior authenticated information. Thus secure key exchange is impossible without authenticated channels. The main question is how much information, *authentic out-of-band messages (OOB messages)*, must be sent over the authenticated channel to achieve reasonable security level. We are aiming at an application, where keys are exchanged between various electronic devices and authentic communication is done by an ordinary user who either enters messages into devices or compares output displays. The latter severely limits a plausible size of OOB messages: one could consider 4–6 decimal digits as optimal and 16 hexadecimal characters as an absolute limit. Other possible OOB channels include various visual or audible signals like blinking lights, images, phone calls etc.

Most urgently such a solution is needed for WLAN: the current use of pre-shared keys degrades both practical usability and security. The home users should have a clear and manageable procedure to set up a secure wireless network so that it is easy to add

and remove devices from the network. Hence, the WiFi Alliance is working on a better solution. Recently, manual data authentication using short authenticated strings received practical applications in ad hoc key agreement. Phil Zimmermann released a software called Zfone and an Internet draft to offer security to Voice over IP [ZJC06]. A similar protocol (See Protocol 3) was adopted by USB-IF for Wireless USB devices [WUS06] and manual data authentication is going to be incorporated into Bluetooth [BT06].

A formal security model for such protocols consists of three bidirectional asynchronous channels, where messages can be arbitrarily delayed. In-band communication is routed from Alice to Bob via an active adversary Charlie, who can drop, modify or insert messages. The out-of-band channel between Alice and Bob is authentic but has low bandwidth and Charlie can arbitrarily delay<sup>3</sup> OOB messages. The model captures nicely all threats in wireless environment, as malicious adversary with a proper equipment can indeed change the network topology and thus reroute, drop, insert and modify messages. However, security is not the only objective. User-friendliness, low resource consumption and simple setup assumptions are equally important. There should be no public key infrastructure, as it is almost impossible to guarantee authenticity and availability of public keys to the humongous number of electronic devices. Also, protocols should use only symmetric primitives if possible.

All currently known user-aided key exchange and data authentication protocols can be divided into two different groups: protocols with authenticated but public OOB messages [Hoe05,CCH06,Vau05,LAN05,PV06a,PV06b,NSS06] and protocols with confidential passwords. Password-protected key exchange, see [BM92,KOY01] and Mana III in [GMN04], is needed when a user wants to establish a secure connection between devices that have input only, for example, devices with keyboards but no display. The main application for the manual data authentication is also a cryptographically secure but still user-friendly ad hoc key agreement between two or more network devices.

**Our contribution.** In this paper, we clarify and extend our preliminary results [LAN05]. In particular, we show that the previously presented manual cross authentication protocols [LAN05,PV06b] are indeed instantiations of the same protocol family that uses a commitment scheme to temporarily hide a secret key needed for data authentication. Compared to the results by Pasini and Vaudenay [PV06b], our security proofs (Sec. 4) are more modular and assumptions on used primitives are weaker and geared towards practice. We explicitly consider implementation details, that is, how to choose practical primitives (Sec. 5). Given a data authentication protocol it can be combined with the Diffie-Hellman key agreement in a secure way by taking the Diffie-Hellman key, or the pair of the public keys, as the data to be authenticated. But the designers of the practical protocols from [ZJC06,WUS06] have taken a different approach by using the Diffie-Hellman key shares as the source of randomness. In Sec. 3, we extend our proof of security also for such a case. In App. A, we consider security in any computational context and show that, under reasonable assumptions, security is not abruptly degraded if several protocols are executed in parallel. As an important theoretical result, we show that all asymptotically optimal (unilateral) manual data authentication protocols have a

---

<sup>3</sup> For example, the adversary can distract the user who compares the output of two devices.

certain structure (App. B, Theorem 5) and that there are no asymptotically optimal two round protocols for data authentication (Corollary 1).

## 2 Cryptographic preliminaries

Our results are formalised in exact security framework where security reductions are precise and thus reveal quantitative differences between various assumptions. Security goals are formalised through games between a challenger and a  $t$ -time adversary<sup>4</sup>  $A$  who tries to violate some design property. Advantage of  $A$  is a non-trivial success probability  $\text{Adv}^{\text{sec}}(A)$  in the game  $\text{sec}$ . The description of  $\text{sec}$  is omitted when the shape of  $\text{Adv}^{\text{sec}}(A)$  reveals the complete structure of  $\text{sec}$ . We consider asymptotic complexity only w.r.t. the time-bound  $t$ . Let  $g(t) = \mathcal{O}(f(t))$  if  $\limsup_{t \rightarrow \infty} g(t)/f(t) < \infty$ . If the working time of adversary can be unbounded, we talk about *statistical security*.

Let  $x \leftarrow \mathcal{X}$  denote independent random draws from a set  $\mathcal{X}$  and  $y \leftarrow A(x_1, \dots, x_n)$  denote assignment according to a randomised algorithm  $A$  with inputs  $x_1, \dots, x_n$ .

**Keyed hash functions.** A keyed hash function  $h : \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{T}$  has two arguments: the first argument corresponds to a data and the second to a key. In our context an applicable tag space  $\mathcal{T}$  is relatively small (may contain as few as  $10^4$  elements) and we need information theoretic properties. A hash function  $h$  is  $\varepsilon_u$ -almost universal, if for any two inputs  $x_0 \neq x_1$ ,  $\Pr[k \leftarrow \mathcal{K} : h(x_0, k) = h(x_1, k)] \leq \varepsilon_u$  and  $\varepsilon_u$ -almost XOR universal if for any  $x_0 \neq x_1$  and  $y$ ,  $\Pr[k \leftarrow \mathcal{K} : h(x_0, k) \oplus h(x_1, k) = y] \leq \varepsilon_u$ .

We need a special notion of almost regular functions when the key is divided into two sub-keys, i.e.,  $h : \mathcal{M} \times \mathcal{K}_a \times \mathcal{K}_b \rightarrow \mathcal{T}$ . A hash function  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular w.r.t. the sub-keys if for each data  $x \in \mathcal{M}$ , tag  $y \in \mathcal{T}$  and sub-keys  $\hat{k}_a \in \mathcal{K}_a, \hat{k}_b \in \mathcal{K}_b$ , we have  $\Pr[k_a \leftarrow \mathcal{K}_a : h(x, k_a, \hat{k}_b) = y] \leq \varepsilon_a$  and  $\Pr[k_b \leftarrow \mathcal{K}_b : h(x, \hat{k}_a, k_b) = y] \leq \varepsilon_b$ . In particular,  $(\varepsilon_a, \varepsilon_b)$ -almost regularity implies that the inequalities hold even if  $y$  is drawn from a distribution that is independent from  $k_a$  and  $k_b$ . Finally, a hash function  $h$  is  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$  if for any two data  $x_0 \neq x_1$  and  $\hat{k}_b, \hat{k}_b \in \mathcal{K}_b$ , we have  $\Pr[k_a \leftarrow \mathcal{K}_a : h(x_0, k_a, \hat{k}_b) = h(x_1, k_a, \hat{k}_b)] \leq \varepsilon_u$ . We say that  $h$  is *strongly  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$*  if for any  $(x_0, k_b) \neq (x_1, \hat{k}_b)$ , we have  $\Pr[k_a \leftarrow \mathcal{K}_a : h(x_0, k_a, k_b) = h(x_1, k_a, \hat{k}_b)] \leq \varepsilon_u$ . Note that  $\varepsilon_u, \varepsilon_a, \varepsilon_b \geq 1/|\mathcal{T}|$  and the word ‘almost’ is skipped in the definitions if the latter equality holds.

**Commitment schemes.** A commitment scheme  $\text{Com}$  is specified by a triple of algorithms (Gen, Com, Open). A setup algorithm Gen generates public parameters  $\text{pk}$  of the commitment scheme. The commitment function  $\text{Com}_{\text{pk}} : \mathcal{M} \times \mathcal{R} \rightarrow \mathcal{C} \times \mathcal{D}$  transforms data  $m \in \mathcal{M}$  into a commitment string  $c$  of fixed length and a decommitment value  $d$ . Usually  $d = (m, r)$ , where  $r \in \mathcal{R}$  is the used randomness. Finally, correctly formed commitments can be opened, i.e.,  $\text{Open}_{\text{pk}}(c, d) = m$  for all  $(c, d) = \text{Com}_{\text{pk}}(m, r)$ . Incorrect decommitment values yield to a special abort value  $\perp$ . We often use a shorthand  $\text{Com}_{\text{pk}}(m)$  to denote  $\text{Com}_{\text{pk}}(m, r)$  with  $r \leftarrow \mathcal{R}$ . Basic properties of commitment

<sup>4</sup> We explicitly assume that adversarial code is executed on a universal Turing or RAM machine.

schemes are defined by hiding and binding games. A commitment scheme is  $(t, \varepsilon_1)$ -hiding if any  $t$ -time adversary  $A$  achieves advantage

$$\text{Adv}_{\text{Com}}^{\text{hid}}(A) = 2 \cdot \left| \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, s \leftarrow \{0, 1\}, (x_0, x_1, \sigma) \leftarrow A(\text{pk}) \\ (c_s, d_s) \leftarrow \text{Com}_{\text{pk}}(x_s) : A(\sigma, c_s) = s \end{array} \right] - \frac{1}{2} \right| \leq \varepsilon_1 .$$

A commitment scheme is  $(t, \varepsilon_2)$ -binding if any  $t$ -time adversary  $A$  achieves advantage

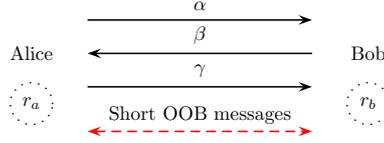
$$\text{Adv}_{\text{Com}}^{\text{bind}}(A) = \Pr \left[ \begin{array}{l} \text{pk} \leftarrow \text{Gen}, (c, d_0, d_1) \leftarrow A(\text{pk}) : \\ \perp \neq \text{Open}_{\text{pk}}(c, d_0) \neq \text{Open}_{\text{pk}}(c, d_1) \neq \perp \end{array} \right] \leq \varepsilon_2 .$$

**Non-malleable commitment schemes.** Many notions of non-malleable commitments have been proposed in cryptographic literature [CIO98,FF00,DG03] starting from the seminal article [DDN91] by Dolev, Dwork and Naor. All these definitions try to capture requirements that are necessary to defeat man-in-the-middle attacks. We adopt the modernised version of [CIO98]—non-malleability w.r.t. opening. The definition is slightly weaker than the definition given in [DG03], as we assume that committed messages are independent from public parameters  $\text{pk}$ . Such choice allows to define non-malleability without a simulator similarly to the framework of non-malleable encryption [BS99].

Intuitively, a commitment scheme is non-malleable, if given a valid commitment  $c$ , it is infeasible to generate related commitments  $c_1, \dots, c_n$  that can be successfully opened after seeing a decommitment value  $d$ . Formally, an adversary is a quadruple  $A = (A_1, A_2, A_3, A_4)$  of efficient algorithms where  $(A_1, A_2, A_3)$  represents an active part of the adversary that creates and afterwards tries to open related commitments and  $A_4$  represents a distinguisher (sometimes referred as a target relation). The adversary succeeds if  $A_4$  can distinguish between two environments  $\text{World}_0$  (real world) and  $\text{World}_1$  (environment where all adversaries are harmless). In both environments, Challenger computes  $\text{pk} \leftarrow \text{Gen}$  and then interacts with adversary  $A$ :

1. Challenger sends  $\text{pk}$  to  $A_1$  that outputs a description of an efficient message sampler  $\text{MGen}$  and a state  $\sigma_1$ . Then Challenger draws two independent samples  $x_0 \leftarrow \text{MGen}$ ,  $x_1 \leftarrow \text{MGen}$  and computes a challenge commitment  $(c, d) \leftarrow \text{Com}_{\text{pk}}(x_0)$ .
2. Challenger sends  $c, \sigma_1$  to  $A_2$  that computes a state  $\sigma_2$  and a commitment vector  $(c_1, \dots, c_n)$  with arbitrary length. If some  $c_i = c$  then Challenger stops  $A$  with  $\perp$ .
3. Challenger sends  $d, \sigma_2$  to  $A_3$  that must produce a *valid* decommitment vector  $(d_1, \dots, d_n)$ . More precisely, Challenger computes  $y_i = \text{Open}_{\text{pk}}(c_i, d_i)$ . If some  $y_i = \perp$  then Challenger stops  $A$  with  $\perp$ .<sup>5</sup>
4. In  $\text{World}_0$  Challenger invokes  $A_4(x_0, y_1, \dots, y_n, \sigma_2)$  with the correct sample  $x_0$  whereas in  $\text{World}_1$  Challenger invokes  $A_4(x_1, y_1, \dots, y_n, \sigma_2)$  with the sample  $x_1$ .

<sup>5</sup> The latter restriction is necessary, as otherwise  $A_3$  can send  $n$  bits of information to  $A_4$  by refusing to open some commitments. The same problem has been addressed [DG03] by requiring that behaviour of  $A_4$  should not change if  $y_i$  is replaced with  $\perp$ . The latter is correct but somewhat cumbersome, as static program analysis of  $A_4$  is undecidable in theory. Also, in a real life protocol a honest party always halts when  $y_i = \perp$  as in our model.



**Fig. 1.** Three round manual authentication protocol

The working time of  $A$  is the total time taken to run  $A_1, \dots, A_4$  and MGen. A commitment scheme is  $(t, \varepsilon)$ -non-malleable iff for any  $t$ -time adversary  $A$  the advantage of distinguishing the two worlds is

$$\text{Adv}_{\text{Com}}^{\text{nm}}(A) = |\Pr[A_4 = 0 | \text{World}_0] - \Pr[A_4 = 0 | \text{World}_1]| \leq \varepsilon .$$

The definition given above is natural in the concrete security framework, as it is conceptually clear and easy to apply. Also, the equivalence result between simulation and comparison based definition of non-malleable encryption [BS99] can be directly generalised.<sup>6</sup> Moreover, the definition of non-malleable encryption is stronger and therefore non-malleable encryption schemes (including CCA2 secure encryption schemes) can be used as non-malleable commitments provided that the public parameters  $\text{pk}$  are generated by the trusted party. See Sec. 5 for more detailed discussion.

### 3 Manual data authentication and key exchange protocols

**Formal security model.** Consider a three round *manual cross-authentication* protocol for data (depicted in Fig. 1) where messages  $\alpha, \beta, \gamma$  are routed via an active adversary Charlie who can drop, delay, modify and insert messages. A low bandwidth out-of-band channel between Alice and Bob is bidirectional and authentic, but Charlie can arbitrarily delay OOB messages. As communication is asynchronous, Charlie can arbitrarily reorder in-band messages, e.g., Bob can receive  $\hat{\alpha}$  from Charlie even before Alice has sent  $\alpha$ . Throughout the article, the hatted messages are received from Charlie and subscripts  $a$  and  $b$  denote values Alice and Bob compute locally. In particular,  $r_a, r_b$  denote random coins and  $m_a, m_b$  input data of Alice and Bob. The desired common output is  $(m_a, m_b)$  if both parties reach accepting state.

We assume that Alice and Bob send two OOB messages  $\text{oob}_{a \rightarrow b}$  and  $\text{oob}_{b \rightarrow a}$  in a fixed order during the protocol. Additionally, we require that the OOB messages have been specified in such a way that either both Alice and Bob accept the output or neither of them does. Often, the second OOB message just indicates whether the sender reached the accepted state. Charlie succeeds in deception if at the end of the protocol Alice and Bob reach the accepting state but  $(m_a, \hat{m}_b) \neq (\hat{m}_a, m_b)$ . A protocol is *correct* if Alice and Bob always reach the accepting state when Charlie does not intervene.

<sup>6</sup> Substitutions in the definitions and proofs of [BS99] are straightforward, except that there is no decommitment oracle and an isolated sub-adversary  $A_3$  has to compute decommitment values.

Let  $A$  be an adversarial algorithm used by Charlie. Then the advantage of  $A$  against data authentication protocol is defined as

$$\text{Adv}^{\text{forge}}(A) = \max_{m_a, m_b} \Pr[\text{Alice and Bob accept } (m_a, \hat{m}_b) \neq (\hat{m}_a, m_b)]$$

where probability is taken over random coins of  $A$  and the honest participants. An authentication protocol is  $(t, \varepsilon)$ -secure if for any  $t$ -time adversary  $A$ ,  $\text{Adv}^{\text{forge}}(A) \leq \varepsilon$ . We use the same adversarial model for user-aided key exchange protocols. Here, the number of exchanged messages might be larger than three and the desired common output consists of a fresh key  $k$  and a unique session identifier  $\text{sid}$ . A key exchange protocol is  $(\varepsilon, t)$ -immune against active attacks if, for any  $t$ -time adversary  $A$ ,

$$\text{Adv}^{\text{forge}}(A) = \Pr[\text{Alice and Bob accept } (\text{sid}, \text{key}_a) \neq (\text{sid}, \text{key}_b)] \leq \varepsilon .$$

A  $(\varepsilon, t)$ -immune key exchange protocol is secure when it can resist passive attacks:  $\varepsilon$  quantifies the maximal security drop against active compared to passive attacks.

Clearly, the protocol outcome is determined by the first OOB message. Moreover, for the  $\ell$ -bit message there exists an efficient deception strategy that with  $\text{Adv}^{\text{forge}}(A) = 2^{-\ell}$ . A protocol family is *asymptotically optimal* if it is possible to choose sub-primitives so that security level reaches asymptotically  $2^{-\ell}$ , see App. B for further discussion.

Other authors [Vau05, PV06a, PV06b] have used more complex framework [BR93] to define security. Such approach is needed only if consecutive runs of authentication protocols are not statistically independent, i.e., protocols use long-lived authentication keys. In our case, all protocol runs are statistically independent, i.e., given  $m_a$  and  $m_b$  or  $\text{sid}$ , a potential adversary can always perfectly simulate all protocol messages. Therefore, our protocols are secure in any computational context, see App. A.

**New protocol families.** Our new construction for cross-authentication protocols covers all currently known asymptotically optimal protocol families: a construction given by Pasini and Vaudenay [PV06b] and our earlier results [LAN05]. The protocol is depicted in Fig. 2. We explicitly assume that all public parameters are generated correctly by a trusted authority, i.e., we assume the common reference string model. Such assumption is not farfetched, as almost all communication standards provide some public parameters, e.g., descriptions of hash functions.

### Protocol

1. Alice computes  $(c, d) \leftarrow \text{Com}_{\text{pk}}(k_a)$  for random  $k_a \leftarrow \mathcal{K}_a$  and sends  $(m_a, c)$  to Bob.
2. Bob chooses random  $k_b \leftarrow \mathcal{K}_b$  and sends  $(m_b, k_b)$  to Alice.
3. Alice sends  $d$  to Bob, who computes  $k_a \leftarrow \text{Open}_{\text{pk}}(c, d)$  and halts if  $k_a = \perp$ .  
Both parties compute a test value  $\text{oob} = h(m_a || m_b, k_a, k_b)$  from the received messages.
4. Both parties accept  $(m_a, m_b)$  iff the local  $\ell$ -bit test values  $\text{oob}_a$  and  $\text{oob}_b$  coincide.

**Specification:**  $h$  is a keyed hash function with sub-keys  $k_a, k_b$  where  $\mathcal{K}_a$  is a message space of commitment scheme. The hash function  $h$  and the public parameters  $\text{pk}$  of the commitment scheme are fixed and distributed by a trusted authority.

**Fig. 2.** Three round cross-authentication protocol Mana IV with  $\ell$ -bit OOB messages.

**Protocol**

1. Alice computes  $(c, d) \leftarrow \text{Com}_{\text{pk}}(k_a)$  for  $k_a = g^a$ ,  $a \leftarrow \mathbb{Z}_q$  and sends  $(\text{id}_a, c)$  to Bob.
2. Bob computes  $k_b = g^b$  for random  $b \leftarrow \mathbb{Z}_q$  and sends  $(\text{id}_b, k_b)$  to Alice.
3. Alice sends  $d$  to Bob, who computes  $k_a \leftarrow \text{Open}_{\text{pk}}(c, d)$  and halts if  $k_a = \perp$ .  
Both parties compute  $\text{sid} = (\text{id}_a, \text{id}_b)$  and  $\text{oob} = h(\text{sid}, k_a, k_b)$  from the received messages.
4. Both parties accept key  $= (g^a)^b = (g^b)^a$  iff the  $\ell$ -bit test values  $\text{oob}_a$  and  $\text{oob}_b$  coincide.

**Specification:**  $h$  is a keyed hash function with sub-keys  $k_a, k_b \in G$  where  $G = \langle g \rangle$  is a  $q$  element Decisional Diffie-Hellman group;  $G$  is a message space of commitment scheme. Public parameters  $\text{pk}$  and  $G$  are fixed and distributed by a trusted authority. Device identifiers  $\text{id}_a$  and  $\text{id}_b$  must be unique in time, for example, a device address followed by a session counter.

**Fig. 3.** Manually authenticated Diffie-Hellman protocol MA–DH with  $\ell$ -bit OOB messages

The Bluetooth authentication mechanisms are undergoing the standardisation phase and the current proposal for the standard [BT06] includes an instantiation of Mana IV (NUMERIC COMPARISON) among other methods. Our security analysis provides the necessary theoretical validation (A more detailed discussion is given in Sec. 5).

One can use the Mana IV protocol to authenticate the transcript of the classical Diffie-Hellman key exchange and thus prevent active attacks. Another reasonable alternative, proposed by Zimmermann and Wireless-USB standard group, is to fuse both protocols into a single one (See Fig. 3). Such solution reduces the number of random bits and computational complexity. Both are scarce resources in small electronic devices. The MA–DH protocol does not directly correspond to these protocols, as it uses commitments to hide  $g^a$  whereas these practical protocols use a cryptographic hash function  $\mathcal{H}$  instead and set  $c = \mathcal{H}(g^a)$ . As a result our security proofs do not directly apply for protocols [ZJC06, WUS06]. Still the results give a some insight and provide suggestions how to achieve provable security (See Sec. 5).

**Related work.** The protocols by Pasini and Vaudenay [PV06b, Fig. 2 and 4] do not directly follow the structure of Mana IV, since in their first message  $\alpha = c$  where  $(c, d) = \text{Com}_{\text{pk}}(m_a || r_a)$  and  $r_a \leftarrow \mathcal{K}_a$ . In our security model, we can always assume that Charlie knows  $m_a$ , as  $m_a$  can be hardwired into the adversarial code. Therefore, if we send  $\alpha = (m_a, c)$ , the security level does not drop and sending  $m_a$  under the commitment becomes unnecessary. As the authenticated data  $m_a$  can be many kilobytes long, it also increases the message space for the commitment scheme. The latter can significantly decrease efficiency, as all currently known provably secure non-malleable commitment schemes are based on asymmetric cryptography.

A modified scheme with  $(c, d) \leftarrow \text{Com}_{\text{pk}}(r_a)$  and  $\alpha = (m_a, c)$  is a specific instance of Mana IV. We also note that in the security proofs of [Vau05, PV06b] it is assumed that the commitment is either a *simulation sound trapdoor commitment scheme* or a hiding one, even if adversary is allowed to query values for non-challenge commitments  $c \neq c_s$ . Both of these notions imply non-malleability [MY04], hence our assumptions are weaker. Moreover, in Sec. 4, we show that non-malleability of  $\text{Com}$  is also necessary, in a certain sense, to the security of the protocol. Finally, a secure fusion of [PV06b] and Diffie-Hellman key exchange similarly to MA–DH becomes problematic (See Sec. 5).

## 4 Security analysis of Mana IV and MA–DH protocols

The structure of Mana IV and MA–DH protocols forces adversary, Charlie, to fix data  $\widehat{m}_a$  and  $\widehat{m}_b$  before the complete hash key  $(k_a, k_b)$  becomes public. Hence, Charlie must either directly attack the hash function  $h$  or some property of commitment scheme to get extra knowledge about the hash key. A good message authentication code  $h$  provides security against simple substitution attacks and basic properties of commitment scheme along with non-malleability safeguard against more complex attacks.

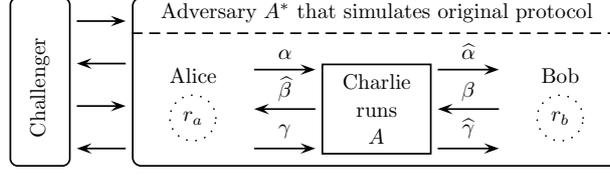
**Theorem 1 (Statistically binding commitments).** *For any  $t$ , there exists  $\tau = t + \mathcal{O}(1)$  such that if  $Com$  is  $(\tau, \varepsilon_1)$ -hiding,  $\varepsilon_2$ -binding and  $(\tau, \varepsilon_3)$ -non-malleable and  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular and  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$  then the Mana IV protocol is  $(2\varepsilon_1 + 2\varepsilon_2 + \varepsilon_3 + \max\{\varepsilon_a, \varepsilon_b, \varepsilon_u\}, t)$ -secure. If additionally  $h$  is also strongly  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$ , then the MA–DH protocol is  $(2\varepsilon_1 + 2\varepsilon_2 + \varepsilon_3 + \max\{\varepsilon_a, \varepsilon_b, \varepsilon_u\}, t)$ -immune against active attacks.*

**Theorem 2 (Computationally binding commitments).** *For any  $t$ , there exists  $\tau = 2t + \mathcal{O}(1)$  such that if  $Com$  is  $(\tau, \varepsilon_1)$ -hiding,  $(\tau, \varepsilon_2)$ -binding and  $(\tau, \varepsilon_3)$ -non-malleable and  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular and  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$  then the Mana IV protocol is  $(2\varepsilon_1 + \varepsilon_2 + \sqrt{\varepsilon_2} + \varepsilon_3 + \max\{\varepsilon_a, \varepsilon_b, \varepsilon_u\}, t)$ -secure. If additionally  $h$  is also strongly  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$ , then the MA–DH protocol is  $(2\varepsilon_1 + \varepsilon_2 + \sqrt{\varepsilon_2} + \varepsilon_3 + \max\{\varepsilon_a, \varepsilon_b, \varepsilon_u\}, t)$ -immune against active attacks.*

*Proof.* For clarity, the proof is split into Lemmata 1–5, as all (including passive) attacks can be divided into four disjoint classes. Combining the corresponding upper bounds on success probabilities proves the claims.  $\square$

Theorems 1 and 2 have several noteworthy implications. First, the Mana IV and MA–DH protocols are indeed asymptotically optimal, see Def. 1 in App. B, as one can choose  $h$  such that  $\max\{\varepsilon_a, \varepsilon_b, \varepsilon_u\} = 2^{-\ell}$  and under standard complexity assumptions there exist commitment schemes where  $\varepsilon_1, \varepsilon_2, \varepsilon_3$  are negligible w.r.t. the security parameter if allowed working time  $\tau$  is polynomial. Secondly, statistically binding commitments give better security guarantee than computationally binding ones:  $\varepsilon_2$  vs.  $\sqrt{\varepsilon_2}$ . The latter is caused by the “non-trivial” reduction technique in Lemma 5. Thirdly, the slight difference in security objectives of Mana IV and MA–DH protocol manifests itself as an extra requirement to  $h$ . This is quite natural: if  $m_a = \widehat{m}_a, m_b = \widehat{m}_b$  but  $(k_a, \widehat{k}_b) \neq (\widehat{k}_a, k_b)$ , we get a correct output for Mana IV but incorrect output for MA–DH, as  $\text{sid}_a = \text{sid}_b$  but  $\text{key}_a \neq \text{key}_b$ . Finally, if Decisional Diffie-Hellman assumption holds for  $G$ , then MA–DH is approximately  $2^{-\ell}$  secure key exchange protocol.

We give a formal security proof of Mana IV and MA–DH by constructing black box reductions corresponding to four different attack types. These reductions have the following structure: given an adversary  $A$  that is good in deception, we construct an adversary  $A^*$  that breaks some property of the commitment scheme. Generic construction of  $A^*$  is depicted on Fig. 4: in order to win a security game  $A^*$  simulates the original protocol and communicates with Challenger. As the communication is asynchronous,  $A$  can reorder protocol messages  $\alpha, \beta, \gamma$ . Let  $\text{msg}_1 \prec \text{msg}_2$  denote that  $\text{msg}_1$  was output on a communication channel before  $\text{msg}_2$ . As Alice and Bob are honest, temporal restrictions  $\alpha \prec \widehat{\beta} \prec \gamma$  and  $\widehat{\alpha} \prec \beta \prec \widehat{\gamma}$  hold for all executions.



**Fig. 4.** Generic reduction scheme

An execution path is almost normal (denoted as norm) if the second round is completed before  $A$  starts the third round, i.e.,  $\alpha, \hat{\alpha}, \beta, \hat{\beta} \prec \gamma, \hat{\gamma}$ . Otherwise, one of the mutually exclusive events  $\gamma \prec \beta$  or  $\hat{\gamma} \prec \hat{\beta}$  must occur. For brevity, let d-forge denote that Alice and Bob accept  $(m_a, \hat{m}_b) \neq (\hat{m}_a, m_b)$  in the Mana IV protocol and k-forge denote that Alice and Bob accept  $(id_a, \hat{id}_b, key_a) \neq (\hat{id}_a, id_b, key_b)$  in the MA-DH protocol. Note that all probabilities in Lemmata 1–5 are taken over random coins of Gen,  $A$  and Alice and Bob and for a fixed input data  $(m_a, m_b)$  or identifiers  $(id_a, id_b)$ . As all proofs are quite straightforward but tedious, only the proof of Lemma 1 covers all details. All other proofs are more compact: some elementary steps are left to the reader.

**Attacks based on almost normal execution paths.** In the simplest attack, Charlie attacks directly  $h$  by altering only  $m_a, m_b, k_b$  and possibly  $\gamma$ . Charlie’s aim here is to cleverly choose  $\hat{k}_b$  so that  $oob_a = oob_b$ . An attack where  $k_b \neq \hat{k}_b$  but other messages are unaltered can be successful against MA-DH but not against Mana IV. Strong  $\varepsilon_u$ -universality w.r.t the sub-key  $k_a$  provides appropriate protection against such attacks.

**Lemma 1.** *For any  $t$ , there exists  $\tau = t + \mathcal{O}(1)$  such that if  $Com$  is  $(\tau, \varepsilon_1)$ -hiding and  $(\tau, \varepsilon_2)$ -binding and  $h$  is  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$ , then for any  $t$ -time adversary  $A$  and input data  $(m_a, m_b)$*

$$\Pr[\text{d-forge} \wedge \text{norm} \wedge c = \hat{c}] \leq \varepsilon_u \cdot \Pr[\text{norm} \wedge c = \hat{c}] + \varepsilon_1 + \varepsilon_2 . \quad (1)$$

*If additionally  $h$  is strongly  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$ , then for any pair of identifiers  $(id_a, id_b)$*

$$\Pr[\text{k-forge} \wedge \text{norm} \wedge c = \hat{c}] \leq \varepsilon_u \cdot \Pr[\text{norm} \wedge c = \hat{c}] + \varepsilon_1 + \varepsilon_2 . \quad (2)$$

*Proof.* ANALYSIS OF MANA IV. Assume a  $t$ -time algorithm  $A$  violates (1). Then  $\Pr[\text{d-forge} \wedge \text{norm} \wedge c = \hat{c} \wedge k_a = \hat{k}_a] \geq \Pr[\text{d-forge} \wedge \text{norm} \wedge c = \hat{c}] - \varepsilon_2$ , or otherwise Alice and  $A$  together can open the commitment  $c$  to two different values  $k_a \neq \hat{k}_a$  with probability more than  $\varepsilon_2$ . The latter contradicts  $(\tau, \varepsilon_2)$ -binding for  $\tau = t + \mathcal{O}(1)$ .

Next, we construct  $A^*$  that wins the hiding game, i.e., given  $pk$  outputs  $(x_0, x_1, \sigma)$  and afterwards given a commitment  $c_s$  for  $s \leftarrow \{0, 1\}$ , can correctly guess the bit  $s$ . The adversary  $A^*$  acts in the following way:

1. Given  $pk$ , chooses  $k_a, k_a^* \leftarrow \mathcal{K}_a$  as  $(x_0, x_1)$  and sends  $(k_a, k_a^*, pk)$  to Challenger.

2. When Challenger replies  $c_s$  for  $(c_s, d_s) = \text{Comp}_{\text{pk}}(x_s)$ ,  $A^*$  simulates a faithful execution of Mana IV with  $\alpha = (m_a, c_s)$  until  $A$  queries  $\gamma$ .  $A^*$  stops the simulation and halts with  $\perp$ , if there is a protocol failure,  $\neg\text{norm}$  or  $c \neq \widehat{c}$ .
3. If  $h(m_a || \widehat{m}_b, k_a, \widehat{k}_b) = h(\widehat{m}_a || m_b, k_a, k_b)$  and  $(m_a, \widehat{m}_b) \neq (\widehat{m}_a, m_b)$  outputs a guess  $s = 0$ , else outputs a guess  $s = 1$ .

Now, consider when the simulation diverges from the real run of Mana IV with the same randomness  $r_a$  and  $r_b$ . If  $s = 0$  then  $(c_0, d_0) = \text{Comp}_{\text{pk}}(k_a)$  and Step 3 does not reflect the protocol outcome in three disjoint cases: (a) abnormal execution or  $c \neq \widehat{c}$ , (b)  $\widehat{\gamma}$  is not a valid decommitment (d-forge does not happen) and (c)  $k_a \neq \widehat{k}_a$ . Therefore, we get  $\Pr[A^* = 0 | s = 0] \geq \Pr[\text{d-forge} \wedge \text{norm} \wedge c = \widehat{c} \wedge k_a = \widehat{k}_a]$ . For  $s = 1$ , we get  $\Pr[A^* \neq \perp | s = 1] = \Pr[\text{norm} \wedge c = \widehat{c}]$ , as simulation is perfect until  $A$  queries  $\gamma$ . Since  $c_1$  and  $k_a$  are statistically independent, all values computed by  $A$  are independent from  $k_a$  and thus  $\Pr[A^* = 0 | s = 1, A^* \neq \perp] \leq \varepsilon_u$ . We arrive at a contradiction, as these bounds imply  $\text{Adv}^{\text{hid}}(A^*) = |\Pr[A^* = 0 | s = 0] - \Pr[A^* = 0 | s = 1]| > \varepsilon_1$  and  $A^*$  runs in time  $t + \mathcal{O}(1)$ .

ANALYSIS OF MA–DH. Lets update only the forgery test in the last step of  $A^*$ :

3. If  $h(\text{id}_a || \widehat{\text{id}}_b, k_a, \widehat{k}_b) = h(\widehat{\text{id}}_a || \text{id}_b, k_a, k_b)$  and  $(\text{id}_a, \widehat{\text{id}}_b, \widehat{k}_b) \neq (\widehat{\text{id}}_a, \text{id}_b, k_b)$  output a guess  $s = 0$ , else output a guess  $s = 1$ .

Similarly to Mana IV,  $\Pr[A^* = 0 | s = 0] \geq \Pr[\text{k-forge} \wedge \text{norm} \wedge c = \widehat{c} \wedge k_a = \widehat{k}_a]$  and  $\Pr[A^* \neq \perp | s = 1] = \Pr[\text{norm} \wedge c = \widehat{c}]$ , since the remaining code of  $A^*$  is identical. The new forgery test forces a restriction  $(x_0, k_b) \neq (x_1, \widehat{k}_b)$  instead of  $x_0 \neq x_1$  and we need strongly  $\varepsilon_u$ -universal  $h$  to bound  $\Pr[A^* = 0 | s = 1, A^* \neq \perp] \leq \varepsilon_u$ .  $\square$

*Note 1.* Strong  $\varepsilon_u$ -universality is necessary for the security of the MA–DH protocol, see Sec. 5 for a concrete counter example.

Another alternative is a direct attack against non-malleability where  $A$  tries to create “cleverly” related sub-keys  $k_a$  and  $\widehat{k}_a$  to bypass the security check.

**Lemma 2.** *For any  $t$ , there exists  $\tau = t + \mathcal{O}(1)$  such that if  $\text{Com}$  is  $(\tau, \varepsilon_3)$ -non-malleable and  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular, then for any  $t$ -time adversary  $A$  and inputs  $(m_a, m_b)$  or session identifier  $(\text{id}_a, \text{id}_b)$*

$$\Pr[\text{d-forge} \wedge \text{norm} \wedge c \neq \widehat{c}] \leq \varepsilon_a \cdot \Pr[\text{norm} \wedge c \neq \widehat{c}] + \varepsilon_3, \quad (3)$$

$$\Pr[\text{k-forge} \wedge \text{norm} \wedge c \neq \widehat{c}] \leq \varepsilon_a \cdot \Pr[\text{norm} \wedge c \neq \widehat{c}] + \varepsilon_3. \quad (4)$$

*Proof.* Let  $A$  be a  $t$ -time algorithm that violates (3). Then we can build an adversary  $A^* = (A_1^*, A_2^*, A_3^*, A_4^*)$  that can break non-malleability of the commitment scheme:

1. Given  $\text{pk}$ ,  $A_1^*$  outputs a uniform sampler over  $\mathcal{K}_a$  and a state  $\sigma_1 = (\text{pk}, m_a, m_b)$ . Challenger computes  $x_0, x_1 \leftarrow \mathcal{K}_a$  and  $(c, d) \leftarrow \text{Comp}_{\text{pk}}(x_0)$ .
2. Given  $c, \sigma_1$ ,  $A_2^*$  simulates the protocol with  $k_b \leftarrow \mathcal{K}_b$  and stops before  $A$  demands  $\gamma$ .  $A^*$  stops the simulation and halts with  $\perp$ , if there is a protocol failure,  $\neg\text{norm}$  or  $c = \widehat{c}$ . Otherwise,  $A_2^*$  outputs a commitment  $\widehat{c}$  and  $\sigma_2$  containing enough information to resume the simulation including  $(m_a, \widehat{m}_a, m_b, \widehat{m}_b, k_b, \widehat{k}_b)$ .

3. Given  $d, \sigma_2$ ,  $A_3^*$  resumes the simulation and outputs  $\widehat{d}$  as a decommitment value.
4. If  $A_3^*$  was successful in opening  $\widehat{c}$  then  $A_4^*(x_s, y, \sigma_2)$  sets  $k_a \leftarrow x_s$  and  $\widehat{k}_a \leftarrow y$  and computes  $\text{oob}_a = h(m_a || \widehat{m}_b, k_a, \widehat{k}_b)$  and  $\text{oob}_b = h(\widehat{m}_a || m_b, \widehat{k}_a, k_b)$ . If  $\text{oob}_a = \text{oob}_b$  but  $(m_a, \widehat{m}_b) \neq (\widehat{m}_a, m_b)$ , then  $A_4^*$  outputs a guess  $s = 0$ , else outputs 1.

Again, consider where the simulation can diverge from the real execution of Mana IV. In both worlds, we can have a discrepancy if execution is abnormal or  $c = \widehat{c}$ . In  $\text{World}_0$ , Step 4 provides a perfect simulation whereas in  $\text{World}_1$   $k_a$  is independent of all variables computed by  $A$ . Therefore, using same methodology as before

$$\begin{aligned} \Pr[A_4^* = 0 | \text{World}_0] &= \Pr[\text{d-forge} \wedge \text{norm} \wedge c \neq \widehat{c}] , \\ \Pr[A_4^* = 0 | \text{World}_1] &\leq \varepsilon_a \cdot \Pr[\text{norm} \wedge c \neq \widehat{c}] , \end{aligned}$$

as  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular. A contradiction as  $\text{Adv}^{\text{nm}}(A^*) > \varepsilon_3$ . For the MA-DH protocol, we have to refine the forgery test in Step 4 similarly to the proof of Lemma 1, but otherwise the analysis is exactly the same.  $\square$

*Note 2.* Obviously, non-malleability w.r.t. every target relation is not necessary. In particular, if  $h$  is fixed then it is necessary and sufficient that  $\text{Com}$  is secure for all adversaries having the same structure as in Lemma 2. The latter requirement is weaker than complete non-malleability, however, one has to reconsider the condition if  $h$  is substituted with a different function and technically such condition is not easier to prove.

**Attacks based on abnormal execution paths.** The remaining two attack patterns are easy to analyse, since they are direct attacks against binding and hiding properties. If  $\widehat{\gamma} \prec \widehat{\beta}$  then successful  $A$  can predict  $k_a$  given only  $c$  and thus win the hiding game.

**Lemma 3.** *For any  $t$  there exists  $\tau = t + \mathcal{O}(1)$  such that if  $\text{Com}$  is  $(\tau, \varepsilon_1)$ -hiding,  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular. Then for any  $t$ -time adversary  $A$  and input  $(m_a, m_b)$  or session identifier  $(\text{id}_a, \text{id}_b)$*

$$\Pr[\text{d-forge} \wedge \widehat{\gamma} \prec \widehat{\beta}] \leq \varepsilon_1 + \varepsilon_a \cdot \Pr[\widehat{\gamma} \prec \widehat{\beta}] , \quad (5)$$

$$\Pr[\text{k-forge} \wedge \widehat{\gamma} \prec \widehat{\beta}] \leq \varepsilon_1 + \varepsilon_a \cdot \Pr[\widehat{\gamma} \prec \widehat{\beta}] . \quad (6)$$

*Proof.* Let  $A$  be a  $t$ -time adversary that violates (5). If  $\widehat{\gamma} \prec \widehat{\beta}$ , the Bob's control value  $\text{oob}_b$  is fixed before  $A$  receives  $\gamma$ . Consider  $A^*$  that plays the hiding game:

1. Given  $\text{pk}$ , chooses  $k_a, k_a^* \leftarrow \mathcal{K}_a$  as  $(x_0, x_1)$  and sends  $(k_a, k_a^*, \text{pk})$  to Challenger.
2. When Challenger replies  $c_s$  for  $(c_s, d_s) = \text{Com}_{\text{pk}}(x_s)$ ,  $A^*$  simulates a faithful execution of Mana IV with  $\alpha = (m_a, c_s)$  until  $A$  outputs  $\widehat{\beta}$ .  $A^*$  stops the simulation and halts with  $\perp$ , if there is a protocol failure,  $\widehat{\gamma} \not\prec \widehat{\beta}$  or  $\text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}) = \perp$ .
3. Next  $A^*$  computes  $\widehat{k}_a = \text{Open}_{\text{pk}}(\widehat{c}, \widehat{d})$ ,  $\text{oob}_a = h(m_a || \widehat{m}_b, k_a, \widehat{k}_b)$  and  $\text{oob}_b = h(\widehat{m}_a || m_b, \widehat{k}_a, k_b)$ . If  $\text{oob}_a = \text{oob}_b$  and  $(m_a, \widehat{m}_b) \neq (\widehat{m}_a, m_b)$  outputs 0, else 1.

Again, consider where the simulation can diverge from the real protocol. If  $s = 0$  then only  $\widehat{\gamma} \not\prec \widehat{\beta}$  can cause the difference. For  $s = 1$ , simulation is perfect until  $\gamma$  is queried

and thus  $\Pr[A^* \neq \perp | s = 1] = \Pr[\widehat{\gamma} \prec \widehat{\beta}]$ . As  $k_a$  is independent from  $\text{oob}_b$ ,  $\widehat{m}_b$  and  $\widehat{k}_b$ , then  $\Pr[A^* = 0 | s = 1, A^* \neq \perp] \leq \varepsilon_a$  follows from  $(\varepsilon_a, \varepsilon_b)$ -almost regularity. A contradiction, as  $\text{Adv}^{\text{hid}}(A^*) > \Pr[\text{d-forge} \wedge \widehat{\gamma} \prec \widehat{\beta}] - \varepsilon_a \cdot \Pr[\widehat{\gamma} \prec \widehat{\beta}] > \varepsilon_1$ . Same algorithm with a redefined forgery check is suitable for the MA–DH protocol.  $\square$

To win the remaining case  $\gamma \prec \beta$ , adversary  $A$  must double open  $\widehat{c}$  to succeed. For statistically binding commitments, the reduction is simple. Analysis of computational binding commitments is more complex.

**Lemma 4.** *If  $\text{Com}$  is statistically  $\varepsilon_2$ -binding and  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular, then for any adversary  $A$  and input  $(m_a, m_b)$  or session identifier  $(\text{id}_a, \text{id}_b)$*

$$\Pr[\text{d-forge} \wedge \gamma \prec \beta] \leq \varepsilon_2 + \varepsilon_b \cdot \Pr[\gamma \prec \beta] , \quad (7)$$

$$\Pr[\text{k-forge} \wedge \gamma \prec \beta] \leq \varepsilon_2 + \varepsilon_b \cdot \Pr[\gamma \prec \beta] . \quad (8)$$

*Proof.* For each commitment  $\widehat{c}$ , fix a canonical  $\widehat{k}_a$  such that  $\widehat{k}_a = \text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}_0)$  for some  $\widehat{d}_0$ . If  $\gamma \prec \beta$  then  $\text{oob}_a$  is fixed before  $k_b$ . Now, the probability that different  $k_b$  values lead to different valid openings  $\widehat{k}'_a \neq \widehat{k}_a$  is at most  $\varepsilon_2$ . Otherwise, one can find valid double openings  $\text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}_0) \neq \text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}_1)$  just by enumerating all possible protocol runs. Now  $\Pr[k_b \leftarrow \mathcal{K} : \text{oob}_a = h(\widehat{m}_a || m_b, \widehat{k}_a, k_b)] \leq \varepsilon_b$ , as  $k_b$  is independent from  $\widehat{k}_a$  and  $\text{oob}_a$  and thus both claims follow.  $\square$

**Lemma 5.** *For any  $t$  there exists  $\tau = 2t + \mathcal{O}(1)$  such that if  $\text{Com}$  is  $(\tau, \varepsilon_2)$ -binding and  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular, then for any  $t$ -time adversary  $A$  and inputs  $(m_a, m_b)$*

$$\Pr[\text{d-forge} \wedge \gamma \prec \beta] \leq \varepsilon_b \cdot \Pr[\gamma \prec \beta] + \sqrt{\varepsilon_2} , \quad (9)$$

$$\Pr[\text{k-forge} \wedge \gamma \prec \beta] \leq \varepsilon_b \cdot \Pr[\gamma \prec \beta] + \sqrt{\varepsilon_2} . \quad (10)$$

*Proof.* Let  $A$  be a  $t$ -time adversary that violates (9). Consider  $A^*$  that

1. Simulates protocol run until  $A$  queries  $\beta$  and stores  $\widehat{c}$ . Halts if  $\gamma \not\prec \beta$ .
2. Provides  $k_b^0, k_b^1 \leftarrow \mathcal{K}_b$  and outputs  $\widehat{c}$  with the corresponding replies  $\widehat{d}_0$  and  $\widehat{d}_1$ .

For a fixed  $\text{pk}$  and  $\widehat{c}$ , let  $\varepsilon_{\text{pk}, \widehat{c}} = \Pr[\text{d-forge} | \gamma \prec \beta, \text{pk}, \widehat{c}]$  denote the forgery probability w.r.t. a single challenge  $k_b$  at Step 2 and

$$\delta_{\text{pk}, \widehat{c}} = \Pr[\perp \neq \text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}_0) \neq \text{Open}_{\text{pk}}(\widehat{c}, \widehat{d}_1) \neq \perp | \gamma \prec \beta, \text{pk}, \widehat{c}]$$

the success probability at Step 2. Then  $\delta_{\text{pk}, \widehat{c}} \geq \varepsilon_{\text{pk}, \widehat{c}}(\varepsilon_{\text{pk}, \widehat{c}} - \varepsilon_b)$ , since  $h$  is  $(\varepsilon_a, \varepsilon_b)$ -almost regular and  $\text{oob}_b^0$  is fixed before  $k_b^1$ . Using a special case of Jensen's inequality,  $E(X^2) \geq E(X)^2$  for any distribution of  $X$ , we get

$$\begin{aligned} \Pr[\text{success} | \gamma \prec \beta] &= \sum_{\text{pk}, \widehat{c}} \Pr[\text{pk} = \text{Gen}, \widehat{c} | \gamma \prec \beta] (\varepsilon_{\text{pk}, \widehat{c}}^2 - \varepsilon_b \varepsilon_{\text{pk}, \widehat{c}}) \\ &\geq \Pr[\text{d-forge} | \gamma \prec \beta]^2 - \varepsilon_b \Pr[\text{d-forge} | \gamma \prec \beta] . \end{aligned}$$

As  $\Pr[\text{d-forge} | \gamma \prec \beta] > \varepsilon_b$ , we get  $\Pr[\text{success} | \gamma \prec \beta] \geq (\Pr[\text{d-forge} | \gamma \prec \beta] - \varepsilon_b)^2$ . Now from  $\Pr[\gamma \prec \beta] \geq \Pr[\gamma \prec \beta]^2$ , we obtain a contradiction

$$\text{Adv}^{\text{bind}}(A^*) \geq \Pr[\gamma \prec \beta]^2 (\Pr[\text{d-forge} | \gamma \prec \beta] - \varepsilon_b)^2 > \varepsilon_2 .$$

The same proof is valid also for the MA–DH protocol.  $\square$

*Note 3.* There are several alternatives to Lemma 5 that offer various tradeoffs between time  $\tau$  and  $\varepsilon_2$  depending how many times  $A$  is probed with different values of  $k_b$ . As  $A$  may totally break the Mana IV protocol on  $\varepsilon_2$  fraction public parameters  $\text{pk}$  and do nothing for other values of  $\text{pk}$ , we cannot get a better bound than  $\Pr[\text{d-forge} \wedge \gamma \prec \beta] \leq \varepsilon_b \cdot \Pr[\gamma \prec \beta] + \varepsilon_2$  with black-box reductions. In our earlier work [LAN05], we used knowledge extraction techniques to obtain more complex reductions.

*Note 4.* Compared to proofs in [Vau05,PV06b] Lemma 5 seems to be inefficient and cumbersome. However, Vaudenay et al uses a different notion of binding—de facto they postulate Lemma 5 for a certain  $h$  as a security requirement. In asymptotic sense these notions are equivalent (there are polynomial reduction between them), but the exact security framework reveals that their condition is quantitatively much stronger.

In practical applications, commitments are constructed from cryptographic hash functions like SHA-1 and classical binding is more appropriate notion, since it leads directly to collision resistance. Secondly, Vaudenay’s approach does not generalise for more complex constructions of  $h$ .

## 5 Practical Implementation Details

**Security constraints.** Mana IV and MA–DH protocols are secure in any computational context if (a) random values are never reused, (b) protocol outputs are never used before reaching the accepting state, (c) there are no multiple protocol instances between the *same* device pair at any time. Then a single protocol instance has same security guarantees as in Theorems 1 and 2. See App. A for a formal proof and discussion.

**Hash functions.** To instantiate Mana IV and MA–DH protocols, we need hash functions  $h : \mathcal{M} \times \mathcal{K}_a \times \mathcal{K}_b \rightarrow \{0, 1\}^\ell$  that are  $(\varepsilon_a, \varepsilon_b)$ -almost regular and (strongly)  $\varepsilon_u$ -almost universal w.r.t. the sub-key  $k_a$ . In our preliminary work [LAN05], we proposed a construction  $h(m, k_a, k_b) = h_0(m, f(k_a, k_b))$  where  $h_0$  is a  $\varepsilon_u$ -almost universal and  $\varepsilon_a$ -regular and  $f : \mathcal{K}_a \times \mathcal{K}_b \rightarrow \{0, 1\}^m$  is regular w.r.t. sub-keys  $k_a, k_b$  and for any  $k_b \neq \widehat{k}_b$  the distribution of pairs  $(f(k_a, k_b), f(k_a, \widehat{k}_b))$  for  $k_a \leftarrow \mathcal{K}_a$  is statistically  $\delta$ -close to uniform distribution. Then it is straightforward to verify that  $h$  is  $(\varepsilon_a, \varepsilon_a)$ -almost regular and  $\max\{\varepsilon_a + \delta, \varepsilon_u\}$ -almost universal, since for  $k_b \neq \widehat{k}_b$  keys  $f(k_a, k_b)$  are  $f(k_a, \widehat{k}_b)$  almost independent.

As a concrete example let  $f : \{0, 1\}^{2m} \times \{0, 1\}^{2m} \rightarrow \{0, 1\}^m$  be defined as follows:  $f(x_0 || x_1, y) = x_0 y \oplus x_1$  in  $\text{GF}(2^m)$  if  $x_0 \neq 0$  and  $f(0^m || x_1, y) = x_1 \oplus y$  otherwise. Clearly,  $f$  is regular w.r.t. the sub-keys and  $f(x_0, x_1, y_1) \oplus f(x_0, x_1, y_2) = x_0(y_1 \oplus y_2)$  covers  $\text{GF}(2^m) \setminus \{0\}$  when  $y_1 \neq y_2$  and  $x_0 \neq 0$ . Hence,  $f$  is  $(\varepsilon_a, \varepsilon_a)$ -almost regular and  $\max\{2^{-m+1} + \varepsilon_a, \varepsilon_u\}$ -secure. Note that for proper choice of  $m$ ,  $2^{-m+1} \ll 2^{-\ell}$ .

Pasini et al., [PV06b] proposed a construction  $h(m, k_a, k_b) = h_0(m, k_a) \oplus k_b$  where  $h_0$  is  $\varepsilon_u$ -almost XOR universal and  $\varepsilon_a$ -almost regular w.r.t.  $k_a$ . The latter is  $(\varepsilon_a, 2^{-\ell})$ -almost regular and strongly  $\varepsilon_u$ -almost universal. But such construction cannot be used in the MA–DH protocol, as  $k_b$  is typically at least 200 bits long. If we compress  $k_b$  in some manner, i.e., compute  $h(m_a || m_b, k_a, k_b) = h_0(m_a || m_b, k_a) \oplus h_1(m_b, k_b)$  then the resulting hash function is only  $\varepsilon_u$ -almost universal. A malicious adversary can choose

$(m_b, k_b) \neq (m_b, \widehat{k}_b)$  such that  $h_1(m_b, k_b) = h_1(m_b, \widehat{k}_b)$ . Since  $\ell$  is small in practical protocols, such pair can be found in real time and Charlie can indeed break the MA–DH protocol by choosing  $\widehat{k}_b = g^c$  for  $c \leftarrow \mathbb{Z}_q$  in this way. As a result Charlie and Alice share a common key. If Bob is a wireless router, then Charlie has successfully completed the attack, as he can transfer Alice’s communication to Bob using secure channel between himself and Bob. Hence, the proper choice of  $h$  is extremely important.

For practical purposes  $\mathcal{M} = \{0, 1\}^{512}$  is sufficiently big, as one can always use a collision resistant hash functions to compress longer messages. And for such parameters many efficient  $\varepsilon_u$ -almost (XOR) universal and perfect hash functions are known with  $\varepsilon_u \leq 2^{-\ell+1}$  (See [Sti91,BJKS93,NGR05] for some concrete examples).

Some practical proposals [BT06, p. 13, 21] propose use cryptographic hash functions to construct  $h$ . The latter is a plausible though heuristic choice, as long as statistical tests do not reveal a significant deviation from desired parameters  $\varepsilon_a, \varepsilon_b, \varepsilon_u$ . Otherwise, the potential adversary can discover and exploit these weaknesses.

**Non-malleable commitment schemes.** The simplest construction of a non-malleable commitment scheme is based on a CCA2 secure encryption scheme. Let  $\text{Enc}_{\text{pk}} : \mathcal{M} \times \mathcal{R} \rightarrow \mathcal{C}$  be a deterministic encryption rule where  $r \in \mathcal{R}$  denotes randomness used to encrypt a message. Define  $(c, d) \leftarrow \text{Com}_{\text{pk}}(x, r)$  as  $c = \text{Enc}_{\text{pk}}(x, r)$  and  $d = (x, r)$  and  $\text{Open}_{\text{pk}}(c, d) = m$  if  $\text{Enc}_{\text{pk}}(x, r) = c$  and  $\perp$  otherwise. Then the corresponding commitment scheme is non-malleable provided that  $\text{pk}$  is generated by a trusted party. We suggest Cramer-Shoup or Desmedt-Kurosawa encryption schemes [CS98,KD04], as the public key is a random tuple of group elements and can be easily generated without the secret key. RSA-OAEP is also CCA2 secure in a random oracle model [FOPS01]. Nevertheless, the key  $\text{pk}$  must be securely distributed, since *a priori* non-malleability w.r.t.  $\text{pk}$  does not guarantee non-malleability w.r.t. related keys  $\text{pk}_1$  and  $\text{pk}_2$ .

All these constructions are *too inefficient* for small electronic devices and they offer too high levels of security. Recall that  $\ell \lesssim 14$  and thus a commitment scheme should be roughly  $(2^{80}, 2^{-20})$ -non-malleable. Secure distribution of  $\text{pk}$  is another problem. In principle, it can be managed as there is only single public key, but may still not be well accepted for industrial applications. There are commitment schemes that are non-malleable without commonly shared  $\text{pk}$ , but these are very inefficient in practice.

In reality, a cryptographic hash functions like SHA-1 are used instead of commitments, as such constructions are hundred times faster and there are no setup assumptions. Let  $\mathcal{H}$  be a collision resistant hash function. Then the hash commitment is computed as  $(c, d) \leftarrow \text{Com}(x, r)$  with  $c = \mathcal{H}(x||r)$  and  $d = (x, r)$  or, as in HMAC,  $c = \mathcal{H}(r \oplus \text{opad}||\mathcal{H}(r \oplus \text{ipad}||x))$  with  $d = r$  (See [BT06, p. 13] as an example). Both constructions are *a priori* not hiding. We would like to have a provably secure construction. In theory, we could use one-wayness of  $\mathcal{H}$  and define commitment with hard-core bits but this leads to large commitments. Instead, we use Bellare-Rogaway random oracle design principle to heuristically argue that a hash commitment based on the OAEP padding is a better alternative. Recall that the OAEP padding is  $c = \mathcal{H}(s, t)$ ,  $s = (x||0^{k_0}) \oplus g(r)$ ,  $t = r \oplus f(s)$ . The corresponding commitment  $c$  along with  $d = r$  is provably hiding and binding if  $g$  is pseudorandom,  $f$  is random oracle, and  $\mathcal{H}$  is collision resistant. A priori SHA-1 and SHA-512 are not known to

be non-malleable, as it has never been a design goal. On the other hand, the security proof of OAEP [FOPS01] shows CCA2 security (non-malleability) provided that  $\mathcal{H}$  is a partial-domain one-way permutation. More specifically, it should be infeasible to find  $s$  given  $h(s, t)$ ,  $s \leftarrow \mathcal{M}_1, t \leftarrow \mathcal{M}_2$ . The partial one-wayness follows for  $r, t \in \{0, 1\}^{80}$  if we assume that  $\mathcal{H}$  is at least  $(2^{160}, 2^{-20})$ -collision resistant as we can enumerate all possible  $t$  values to get a collision. The other assumption that  $h$  is a permutation is important in the proof. Therefore, we can only *conjecture* that the proof can be generalised and the OAEP padding provides a non-malleable commitment scheme.

Hence, an important theoretical task is to provide efficient but provably hiding and non-malleable but efficient padding construction for hash commitments. Also, one could reprove Lemma 1 and Lemma 3 without assuming hiding from  $Com$ , as in both proofs we do not need hiding of  $k_a$  but just Charlie's inability to control Alice's  $oob_a$ . Practical implementations [ZJC06,WUS06] of the MA-DH protocol use  $c = \mathcal{H}(g^a)$  and such a relaxed security proof would bridge the gap between theory and practice.

**Acknowledgements.** We would like to thank N. Asokan for joint work on the initial solution and for many useful discussions and comments, and Emilia Käsper for helpful suggestions. The first author was partially supported by the Finnish Academy of Sciences and Estonian Doctoral School in Information and Communication Technologies.

## References

- [BJKS93] J. Bierbrauer, T. Johansson, G. Kabatianskii, and B. Smeets. On families of hash functions via geometric codes and concatenation. In *Proc. of CRYPTO '93*, LNCS 773. Springer, 1993.
- [BM92] S. Bellovin and M. Merrit. Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks. In *Proc. of the IEEE Symposium on Security and Privacy*, pages 72–84, 1992.
- [BR93] Mihir Bellare and Phillip Rogaway. Entity Authentication and Key Distribution. In *Proc. of CRYPTO '93*, LNCS 773, pages 232–249. Springer, 1993.
- [BS99] Mihir Bellare and Amit Sahai. Non-malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In *Proc. of CRYPTO '99*, LNCS 1666, pages 519–536. Springer, 1999.
- [BT06] Bluetooth Special Interest Group. Simple Pairing Whitepaper (Revision V10r00). [http://www.bluetooth.com/Bluetooth/Apply/Technology/Research/Simple\\_Pairing.htm](http://www.bluetooth.com/Bluetooth/Apply/Technology/Research/Simple_Pairing.htm), 2006.
- [CCH06] M. Cagalj, S. Capkun, and J.-P. Hubaux. Key agreement in peer-to-peer wireless networks. *Proc. of the IEEE*, 94(2):467–478, Feb 2006.
- [CIO98] Giovanni Di Crescenzo, Yuval Ishai, and Rafail Ostrovsky. Non-interactive and non-malleable commitment. In *STOC '98*, pages 141–150, 1998.
- [CS98] Ronald Cramer and Victor Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *Proc. of CRYPTO '98*, LNCS 1462, pages 13–25. Springer, 1998.
- [DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-malleable cryptography. In *STOC '91*, pages 542–552, New York, NY, USA, 1991. ACM Press.
- [DG03] Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *STOC 2003*, pages 426–437, 2003.

- [FF00] Marc Fischlin and Roger Fischlin. Efficient Non-malleable Commitment Schemes. In *Proc. of CRYPTO 2000*, LNCS 1880, pages 413–431. Springer, 2000.
- [FOPS01] Eiichiro Fujisaki, Tatsuaki Okamoto, David Pointcheval, and Jacques Stern. RSA-OAEP Is Secure under the RSA Assumption. In *Proc. of CRYPTO 2001*, LNCS 2139, pages 260–274, 2001.
- [GMN04] Christian Gehrman, Chris J. Mitchell, and Kaisa Nyberg. Manual authentication for wireless devices. *RSA Cryptobytes*, 7(1):29–37, January 2004.
- [Hoe05] Jaap-Henk Hoepman. Ephemeral Pairing on Anonymous Networks. In *Proc. of SPC 2005*, LNCS 3450, pages 101–116. Springer, 2005.
- [KD04] Kaoru Kurosawa and Yvo Desmedt. A New Paradigm of Hybrid Encryption Scheme. In *Proc. of CRYPTO 2004*, LNCS 3152, pages 426–442. Springer, 2004.
- [KOY01] Jonathan Katz, Rafail Ostrovsky, and Moti Yung. Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords. In *Proc. of EUROCRYPT 2001*, LNCS 2045, pages 475–494. Springer, 2001.
- [LAN05] Sven Laur, N. Asokan, and Kaisa Nyberg. Efficient Mutual Data Authentication Using Manually Authenticated Strings: Preliminary Version. Cryptology ePrint Archive, Report 2005/424, 2005. <http://eprint.iacr.org/>.
- [LN06] Sven Laur and Kaisa Nyberg. Efficient Mutual Data Authentication Using Manually Authenticated Strings: Extended Version. Cryptology ePrint Archive, Report 2005/424, 2006. <http://eprint.iacr.org/>.
- [MY04] Philip D. MacKenzie and Ke Yang. On Simulation-Sound Trapdoor Commitments. In *Proc. of EUROCRYPT*, LNCS 3027, pages 382–400. Springer, 2004.
- [NGR05] Kaisa Nyberg, Henri Gilbert, and Matt Robshaw. Galois MAC with forgery probability close to ideal. General Public Comments on NIST Cryptopage, 2005.
- [NSS06] Moni Naor, Gil Segev, and Adam Smith. Tight Bounds for Unconditional Authentication Protocols in the Manual Channel and Shared Key Models. In *Proc. of CRYPTO 2006*, LNCS 4117, pages 214–231. Springer, 2006.
- [PV06a] Sylvain Pasini and Serge Vaudenay. An Optimal Non-interactive Message Authentication Protocol. In *Proc. of CT-RSA 2006*, LNCS 3860, pages 280–294. Springer, 2006.
- [PV06b] Sylvain Pasini and Serge Vaudenay. SAS-Based Authenticated Key Agreement. In *PKC 2006*, LNCS 3958, pages 395–409. Springer, 2006.
- [Sti91] D. R. Stinson. Universal Hashing and Authentication Codes. In *Proc. of CRYPTO '91*, LNCS 576, pages 74–85. Springer, 1991.
- [Vau05] Serge Vaudenay. Secure Communications over Insecure Channels Based on Short Authenticated Strings. In *Proc. of CRYPTO 2005*, LNCS 3621, pages 309–326. Springer, 2005.
- [WUS06] Association Models Supplement to the Certified Wireless Universal Serial Bus Specification, 2006. <http://www.usb.org/developers/wusb/>.
- [ZJC06] Philip Zimmermann, Alan Johnston, and Jon Callas. ZRTP: Extensions to RTP for Diffie-Hellman Key Agreement for SRTP draft-zimmermann-avt-zrtp-01, March 2006.

## A Security in arbitrary computational context

Assume that Mana IV and MA–DH protocols are run so that the security constraints presented in Sec. 5 are fulfilled. Then a protocol instance is uniquely determined by the time, origin and destination of the OOB message and a potential adversary cannot

interleave OOB messages. This restriction can be trivially fulfilled—there is no need to exchange more than one key at a time and multiple messages can be sent together.

Consider ideal implementations of cross-authentication and Diffie-Hellman key exchange protocols. In ideal world, given  $m_a$  and  $m_b$  adversary can either deliver them to Alice and Bob or drop messages. Similarly, given  $g^a$ ,  $g^b$  and  $\text{sid}$ , adversary can either do a passive attack against the key exchange protocol or interrupt it. Now consider a security game  $\text{sec}$  that defines security of a complex protocol. Next, theorem shows that the security drop compared to the ideal implementation is at most  $\varepsilon$ .

**Theorem 3.** *Let  $t_p$  be the total computational time needed to complete a complex protocol  $\Pi$ . For any  $t$ -time adversary  $A$  such that  $\text{Adv}_{\text{real}}^{\text{sec}}(A) = \delta$  in the real protocol, there exists a  $(t + t_p)$ -time adversary  $A^*$  that achieves  $\text{Adv}_{\text{ideal}}^{\text{sec}}(A^*) \geq \delta - \varepsilon$ , if used Mana IV or MA-DH protocol is at least  $(t + t_p + \mathcal{O}(1), \varepsilon)$ -secure.*

*Proof (Sketch).* Since the source and destination of OOB messages together with the time uniquely reveal the corresponding Mana IV or MA-DH instance, it is straightforward to verify that honest Alice and Bob accept  $m_a || \hat{m}_b \neq \hat{m}_a || m_b$  or  $(\text{sid}_a, \text{key}_a) \neq (\text{sid}_b, \text{key}_b)$  with probability at most  $\varepsilon$ . Otherwise, we can simulate the surrounding computational context and break a stand-alone Mana IV or MA-DH protocol instance with larger probability than  $\varepsilon$ . Of course, the preceding history that determines  $(m_a, m_b)$  or  $\text{sid}$  should be fixed in such attack. As all previously used random coins can be hard-wired, the corresponding attack still takes  $t + t_p + \mathcal{O}(1)$  steps.

Now consider an adversary  $A^*$  that tries to win the game  $\text{sec}$  in the ideal world. It simulates a real protocol run to the adversary  $A$ . Essentially,  $A^*$  provides  $A$  a direct access to the ideal world except for the absent Mana IV or MA-DH protocol instance. Given  $(m_a, m_b)$  or  $(\text{id}_a, g^a, \text{id}_b, g^b)$ ,  $A^*$  simulates the corresponding protocol to  $A$ . If  $A$  succeeds in deception, then  $A^*$  halts. Otherwise it simulates the end result of the protocol in the ideal world, i.e., delivers all messages unaltered or drops them. Note that when  $A^*$  does not halt then there is no discrepancy between the ideal and real protocol run. Since  $\Pr[A^* \text{ halts}] \leq \varepsilon$  due to the first part of the proof, the result follows.  $\square$

*Note 5.* If many protocol instances can be run in parallel between the same device pair, then there are no security guarantees. When more than  $2^\ell$  protocols run in parallel, then Charlie can certainly swap OOB messages so that at least one attacked protocol reaches accepting state. Of course, such attack is not practical.

## B Theoretical limitations

In the following, we show that there are no asymptotically optimal two round manual message authentication protocols. In other words, two round protocols are inherently less secure. However, the exact quantification of such security drop is out of our scope.

Here it is advantageous to consider unilateral authentication protocols since unilateral authentication is a special case of cross authentication. In a manual unilateral authentication protocol Sender wants to transfer a authentic message  $m$  to Receiver. The restrictions to the protocols are same: protocol must be correct, the order of all

messages is fixed ahead and the first OOB message  $\text{oob}$  determines the protocol outcome. Let  $\ell$  the maximal length of  $\text{oob}$ . We explicitly assume that  $\ell \leq 30$ , since for sufficiently large  $\ell$  (say 160 bits) one can use collision resistant hash functions to protect authenticity, e.g., send  $\text{oob} = h(m)$ . We also assume that the length of inputs  $m$  is larger than  $\ell$  or otherwise we can send  $m$  as the first OOB message. Note that a simple collision attack where Charlie interacts honestly but uses  $2^{\ell+1}$  pre-tabulated values for input and randomness gives a formal proof to the “folklore” bound.

**Theorem 4.** *Let  $\pi$  be a correct unilateral authentication protocol with fixed message order and let  $\ell$  be the maximal length of the first out-of-band message. Then there exists a strategy  $A$  such that  $\text{Adv}_{\pi}^{\text{forge}}(A) \geq 2^{-\ell}$ .*

*Proof.* The proof is omitted due to the lack of space. The complete proof is given in the extended version of the current article [LN06].  $\square$

Such strategy is feasible to follow in real-time for  $\ell \leq 30$ , as necessary pre-tabulated values can be hardwired into the code of  $A$  and then the computational resources needed to construct the actual program code are irrelevant.

Next we show that no two round protocols can achieve security bounds arbitrarily close to  $2^{-\ell}$ . A practical protocol must be secure against the attacks that take super-linear time w.r.t. the honest protocol run or otherwise the security margin is too small.

**Definition 1.** *We say that a protocol family  $\{\pi_k\}$  with a fixed message ordering is asymptotically optimal when the maximal advantage  $\varepsilon_k$  with respect to the time-bound  $t_k$  approaches  $\varepsilon_k \rightarrow 2^{-\ell}$  and  $t_k$  is at least super-linear in the protocol complexity.*

In principle, unilateral authentication protocols can have arbitrary structure. If we assume asymptotic optimality from the protocol family, then we can show that for large enough  $k$ ,  $\text{oob}(m, r_r, r_s)$  is almost uniform w.r.t. to Receiver’s randomness  $r_r$  and Sender’s randomness  $r_s$ , and with high probability only a single value  $\text{oob}(m, r_r, r_s)$  leads to acceptance. Formally, we need a concept of uniform convergence to state these properties. A parametrised sequence  $x_k(a)$  converges uniformly  $x_k(a) \rightrightarrows x$  with respect to the parameter  $a \in \{0, 1\}^*$ , if  $\limsup_k \sup_a x_k(a) = \liminf_k \inf_a x_k(a) = x$ .

**Theorem 5.** *Let  $\{\pi_k\}$  be an asymptotically optimal and correct protocol family, let probability  $\Pr[\cdot|\pi_k]$  be taken over honest runs of  $\pi_k$  and let two-oob denote the event that more than one value of  $\text{oob}$  lead to acceptance. Then next statements are true:*

- (a)  $\Pr[\text{oob}|m, r_s, \pi_k] \rightrightarrows 2^{-\ell}$  w.r.t. the parameters  $m, r_s$ ,
- (b)  $\Pr[\text{two-oob}|m, \pi_k] \rightrightarrows 0$  w.r.t. the parameter  $m$ ,
- (c)  $\Pr[\text{oob}|r_r, m, \pi_k] \rightrightarrows 2^{-\ell}$  w.r.t. the parameters  $r_r$  and  $m$ .

*Proof.* Follows from an extended analysis of the collision attack, see [LN06].  $\square$

**Corollary 1.** *There are no asymptotically optimal and correct two round protocol families with a fixed message order for unilateral authentication.*

*Proof.* Omitted, see the extended version of [LN06].  $\square$

The stated result is rather weak, since it does not quantify how close to the optimal bound the deception probability of two round protocols can go. More accurate analysis is still an open question.