UNIVERSITÉ DU LUXEMBOURG

PhD-FSTC-2019-31
The Faculty of Sciences, Technology and Communication

DISSERTATION

Defence held on 14/05/2019 in Esch-sur-Alzette
to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG EN INFORMATIQUE

by

Jorge Luis TORO POZO

Born on 30 October 1988 in Las Tunas (Cuba)

COMPUTATIONAL AND SYMBOLIC ANALYSIS OF DISTANCE-BOUNDING PROTOCOLS

Dissertation defence committee

Dr. Yves Le Traon, Chairman
Professor, Université du Luxembourg

Dr. Rolando Trujillo Rasúa, Vice-Chairman
Deakin University, Australia

Dr. Sjouke Mauw, Dissertation supervisor
Professor, Université du Luxembourg

Dr. Ioana Boureanu
University of Surrey, United Kingdom

Dr. Stéphanie Delaune
IRISA, Rennes, France

DOCTORAL THESIS

Computational and symbolic analysis of distance-bounding protocols

Jorge Luis TORO POZO

Supervisor
Prof. Dr. Sjouke MAUW
Université du Luxembourg, Luxembourg

Scientific Advisor
Dr. Rolando TRUJILLO RASÚA
Deakin University, Australia

2019

The author was supported by Luxembourg Fonds National de la Recherche (FNR) under the grant AFR-PhD-10188265.

Abstract

Contactless technologies are gaining more popularity everyday. Credit cards enabled with contactless payment, smart cards for transport ticketing, NFC-enabled mobile phones, and e-passports are just a few examples of contactless devices we are familiar with nowadays. Most secure systems meant for these devices presume physical proximity between the device and the reader terminal, due to their short communication range. In theory, a credit card should not be charged of an on-site purchase if the card is not up to a few centimeters away from the payment terminal. In practice, this is not always true. Indeed, some contactless payment protocols, such as Visa's payWave, have been shown vulnerable to relay attacks. In a relay attack, a man-in-the-middle uses one or more relay devices in order to make two distant devices believe they are close. Relay attacks have been implemented also to bypass keyless entry and start systems in various modern cars.

Relay attacks can be defended against with distance-bounding protocols, which are security protocols that measure the round-trip times of a series of challenge/response rounds in order to guarantee physical proximity. A large number of these protocols have been proposed and more sophisticated attacks against them have been discovered. Thus, frameworks for systematic security analysis of these protocols have become of high interest. As traditional security models, distance-bounding security models sit within the two classical approaches: the computational and the symbolic models. In this thesis we propose frameworks for security analysis of distance-bounding protocols, within the two aforementioned models.

First, we develop an automata-based computational framework that allows us to generically analyze a large class of distance-bounding protocols. Not only does the proposed framework allow us to straightforwardly deliver computational (in)security proofs but it also permits us to study problems such as optimal trade-offs between security and space complexity. Indeed, we solve this problem for a prominent class of protocols, and propose a protocol solution that is optimally secure amongst space-constrained protocols within the considered class.

Second, by building up on an existing symbolic framework, we develop a causality-based characterization of distance-bounding security. This constitutes the first symbolic property that guarantees physical proximity without modeling continuous time or physical location. We extend further our formalism in order to capture a non-standard attack known as terrorist fraud. By using our definitions and the verification tool Tamarin, we conduct a security survey of over 25 protocols, which include industrial protocols based on the ISO/IEC 14443 standard such as NXP's MIFARE Plus with proximity check and Mastercard's PayPass payment protocol. For the industrial protocols we find attacks, propose fixes and deliver security proofs of the repaired versions.

Acknowledgments

I must thank many people here. I promise I'll do my best to mention as many as possible. Sorry for those who I'll miss.

I'd like first to thank my supervisor Prof. Dr. Sjouke Mauw for his guidance and encouragement during these four years. I remember he told me once "we expect great things from you", which was within the first few months of my PhD. As scary as it sounds, that was a boost of motivation and has pushed me to aim for nothing but the best. It's been a pleasure to be his apprentice.

I would also like to thank my daily advisor and compatriot Dr. Rolando Trujillo, with whom I could only speak in Spanish over a beer, basically (kind of Sjouke's rule). I enjoyed a lot the never-ending, exciting technical meetings in which we would just throw maths on a whiteboard. I can't thank him enough for his personal and professional support since the very first contact, back in 2014. I'll always be grateful *compay*.

The completion of this thesis would've not been possible were it not be for the support of my family. In special, my mother, whom I'll always owe everything to, and Yani, my beautiful wife –*te amo mi chiquita*. I thank my dad and brother too, they have been there for me at all times.

I want to deeply express my gratitude to the rest of the defence committee: Prof. Dr. Yves Le Traon, Dr. Stéphanie Delaune, and Dr. Ioana Boureanu. Thank you all for taking the time to review my thesis, your inputs are highly appreciated. I cannot think of more suited judges.

I want to thank the Luxembourg National Research Fund (FNR) for the financial support.

Last but not least, I want to thank my friends and close colleagues: Aramis, Luis Angel, Chincha, Mary, Sami, Amin, Chuks, Serket, Ryan, Olga, Yunior, Stas, Zach, Alek, Cui, and Samir.

To all and every single one of you, many thanks.

Jorge Luis TORO POZO

Luxembourg, April 2019

Contents

List of Figures

List of Tables

1

Introduction

Contactless technologies have drastically changed our way of communication. Not only have we moved from fixed to mobile telephony, but we are also using more and more contactless devices such as mobile phones enabled with Near Field Communication (NFC) technology, and Radio Frequency Identification (RFID) smart cards. Contactless communication has multiple applications, which include access control, transport ticketing, e-passports, loyalty cards, and in-store payments. Many modern mobile devices, such as phones, tablets, and smart watches, feature NFC interfaces so users can perform on-site transactions without having to carry any traditional wallets.

According to a recent report by Mastercard, consumers can pay nowadays with a simple tap with their Mastercard credit/debit cards in over 8 million point-of-sale (POS) locations across 111 countries. In the UK, for example, one in two Mastercard transactions are contactless. The company also claims that by 2020 all POS terminals in Europe will be enabled for contactless payments. Another giant, Visa, announced in its Fiscal Year 2018's report that Transport for London (TfL) has seen more than 1 billion Visa contactless journeys since launch. The company also reported that in this year, nearly one out of four face-to-face Visa domestic transactions were contactless.

Speed and convenience are the flagship advantages of contactless communication over contact-based. Contact-based communication naturally enforces proximity between the communicating devices, e.g. when a credit card is inserted in a POS terminal, or a mechanic key is inserted into its keyhole. However, such form of "proximity by design" cannot be presumed in a contactless (or wireless) setting as wireless communication technologies can be often tampered with. For example, while RFID- and NFC-based devices have a small communication range (typically a few centimeters), this does not guarantee proximity because such devices can still communicate at arbitrary distances by means of *relay*.

Relay is a technique that repeats a signal to make it reachable beyond the primary signal's coverage area. While relay is a useful technique, e.g. it is the basis for satellite communication, it can be used by a malicious party to abuse proximity-based secure systems. For example, an RFID tag can be activated by placing a concealed, corrupt reader near the tag so the latter falls into the electromagnetic field of the reader. By implementing the same setting between a corrupt tag co-located with a legitimate RFID, and by simply relaying the communication between the two corrupt devices, an attacker could trick the legitimate reader into believing that the legitimate tag is close (see Figure 1.1).

The first reference to a *relay attack* possibly appeared in the so-called Chess Grandmaster

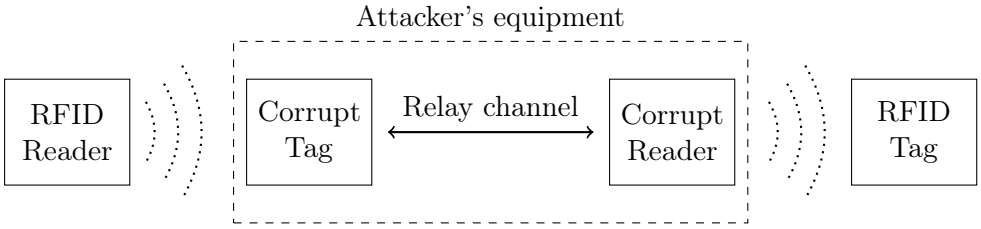


Figure 1.1 A representation of a relay attack, in which the legitimate RFID reader falsely believes that the legitimate RFID tag is within a valid communication range.

problem [Con76]. A little girl who does not know how to play chess manages to win (or at least draw) in one out of two simultaneous matches against each of two chess grandmasters. The strategy employed by the girl consists of simply relaying the moves made by each of her opponents to the other.

Various practical relay attacks have been implemented as well. As reported in [FDC11], Francillon et al. conducted relay attacks on the Passive Keyless Entry and Start (PKES) system of ten cars from eight manufacturers. The authors implemented their attack by simply relaying the communication between the car and the car's legitimate key, located up to 50 meters away from the car. Various other demonstrations of relay attacks on cars can be found on YouTube[1].

1.1 Distance-Bounding Protocols

In 1990, Beth and Desmedt [BD90] proposed the integration of messages' time-of-flight as an extra verification layer into security protocols in order to guarantee physical proximity between the communicating parties, thus preventing relay attacks. This notion was later formalized as *distance-bounding protocols* by Brands and Chaum in [BC93].

Despite the existence of distance-bounding protocols in the academic literature for almost 30 years, they seem to be under-acknowledged when developing secure systems, as the relay attacks described earlier prove. This is often attributed to under-estimation of the damage of relay attacks, and/or the usual assumption that proximity holds by design or is enforced by the environment. However, proximity must be accounted for at the communication protocol level, when designing secure proximity-sensitive systems. There have been undergoing efforts in developing distance-bounding systems by smart cards manufacturers such as the well-known NXP and EMV. We do not intend to uncover the reasons for which distance-bounding protocols have not been widely implemented (yet), though they seem to be gaining popularity.

Distance-bounding protocols aim to securely establish an upper bound on the distance between a *prover* (e.g. an RFID tag) and a *verifier* (e.g. an RFID reader) by measuring the round-trip time (RTT) of a number of rounds of challenges and responses. This round-trip time is the time lapse between the verifier's sending of a challenge and the verifier's receiving of the corresponding response from the prover. More precisely, let us denote by c the (ideally tight) upper bound on the propagation speed of the communication channel (c equals the speed of light for radio waves), and by Δt the RTT of a challenge/response

[1] E.g. https://www.youtube.com/watch?v=0AHSDy6AiV0

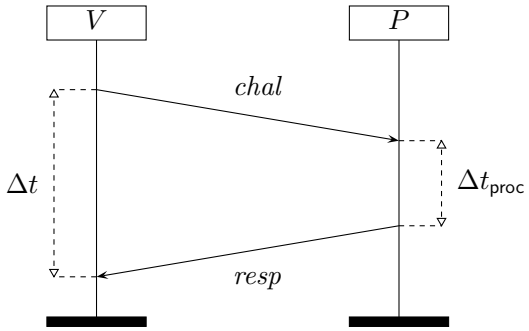


Figure 1.2 A challenge/response round-trip time Δt measurement by a verifier V to upper-bound their distance to a prover P. Timeline goes from top to bottom.

round between a verifier V and a prover P. In addition, let d be the distance between V and P and let Δt_{proc} be the time that the prover takes, upon reception of the challenge, to produce the response. Then:

$$d \leq \frac{\mathsf{c}}{2}(\Delta t - \Delta t_{\mathsf{proc}}) \leq \frac{\mathsf{c}}{2}\Delta t, \tag{1.1}$$

which means that, while d is unknown to V, they can assert that d is *small* by checking that $\Delta t \leq \Delta t_{\mathsf{max}}$, where Δt_{max} is a threshold on the RTT that defines the verifier's proximity. See the Message Sequence Chart (MSC) [CM12] in Figure 1.2 for graphical reference.

A well-known distance-bounding protocol is that of Brands and Chaum [BC93], depicted in Figure 1.3. The protocol consists of three phases. The *slow phase* (a.k.a. initial phase, setup phase, lazy phase) where the parties agree on the parameters of the protocol, such as the number n of rounds of the RTT measurements. In addition, the prover commits to a random sequence of bits $\beta = \beta_1 \cdots \beta_n$ and transmits the sequence encoded. The encoded sequence can be the encryption of β with a freshly generated symmetric key k, which is not revealed at the moment. The verifier stores this commit message for later authentication of the prover. Then the *fast phase* (a.k.a. timed phase, distance-bounding phase, rapid phase, time-measurement phase) starts, composed of n rounds. At the i-th round, the verifier starts a clock for the RTT measurement and sends a random bit-challenge α_i. Upon reception of the challenge, the prover instantly replies with $\alpha_i \oplus \beta_i$. Once the verifier receives the prover's response, the former stops the clock and declares proximity, if the measured round-trip time is not larger than a given threshold Δt_{max}. Authentication, on the other hand, is verified during a *final phase* (a.k.a. verification phase) in which the prover opens the commit (e.g. by revealing k) and signs the fast phase exchanges.

1.2 Security Analysis

This thesis focuses on security analysis of distance-bounding protocols. As traditional security protocol analysis, the analysis techniques will be developed through *computational* and *symbolic* models. Computational models were introduced in the 1980s by a number of works such as [GM84, GMR88, Yao82], whilst symbolic models are attributed to Dolev and Yao [DY83] as well as Needham and Schroeder [NS87].

Both computational and symbolic approaches are supported by mathematical models for the protocol's execution, cryptographic primitives and adversarial behavior. They formally

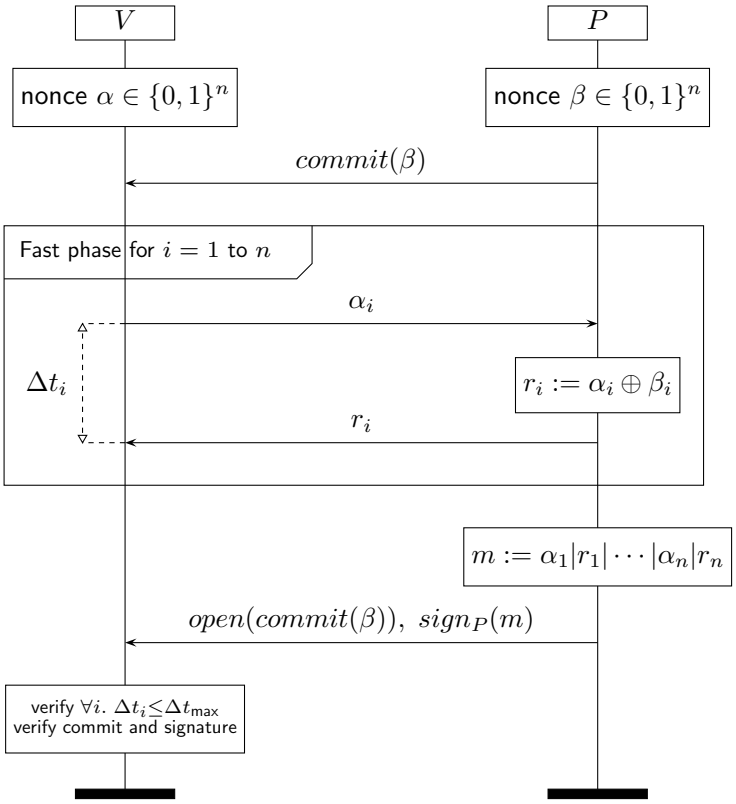
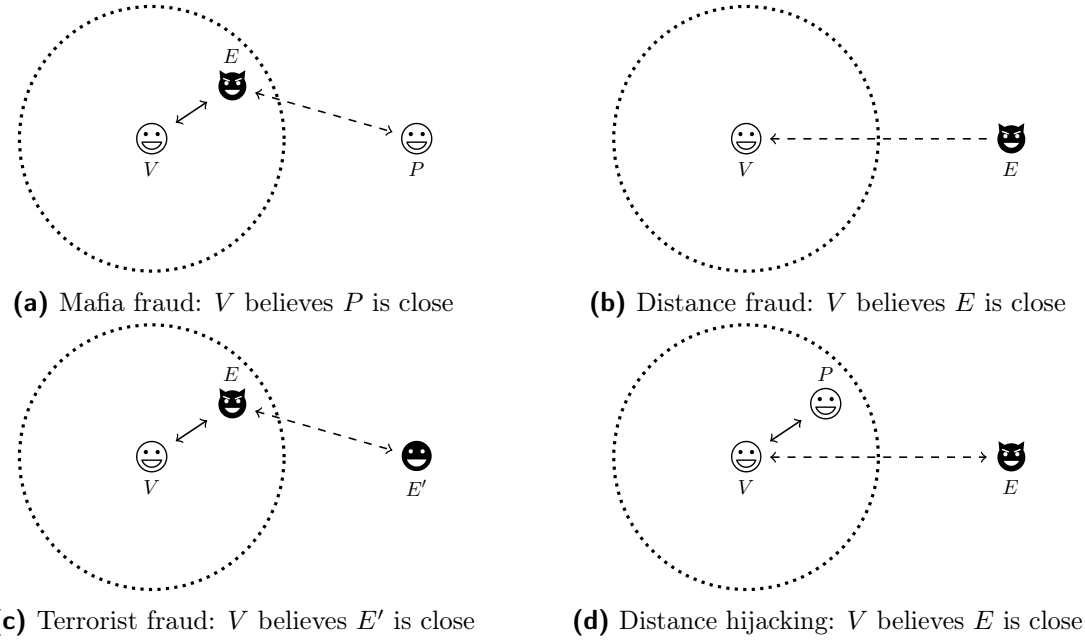


Figure 1.3 Brands and Chaum's (BC) protocol [BC93]. This is the traditional version with signature-based identification scheme.

define security properties expected from cryptographic specifications, and provide methods to rigorously verify that the specified protocols meet these requirements [CKW11]. We do not intend to compare the two approaches. Instead, we will provide a concrete model within each, and show how both proposals can be used to comprehensively construct attacks or deliver security proofs.

In the computational model, the messages are bit-strings, and the cryptographic primitives are functions from bit-strings to bit-strings. Security proofs are assessed against arbitrary probabilistic polynomial-time algorithms (which are the attacks), thus offering strong security guarantees. However, even for moderately lengthy protocols, the proofs are long, tedious, and highly prone to human error, both when delivering them and assessing their soundness as well. In the symbolic model, on the other hand, cryptographic primitives are represented by function symbols considered as absolutely secure black-boxes, messages are symbolic terms on these primitives, and the adversary is restricted to compute only using these primitives. Thus security proofs follow from a high-level, logic analysis of the protocol executions, regardless of the cryptographic algorithms the protocol implements. Often, those proofs are simple, produced and verified (semi-)automatically with tools such as Isabelle/HOL [NPW02], ProVerif [Bla01], and Tamarin [MSCB13].

On the one hand, attacks on a given protocol that exist in the symbolic model also exist in the computational model. On the other hand, attacks on a protocol that exist in the computational model and do not exist in the symbolic model are often the result of the



(a) Mafia fraud: V believes P is close

(b) Distance fraud: V believes E is close

(c) Terrorist fraud: V believes E' is close

(d) Distance hijacking: V believes E is close

Figure 1.4 Types of attack on distance-bounding protocols. In all cases, V is an honest verifier, P is an honest prover, and E is a dishonest prover. The encircled area represents V's proximity, which is bounded by the predefined round-trip threshold. Dashed arrows represent untimed communication, which is communication that does not occur entirely within the fast phase.

adversary's exploitation of arithmetic properties of the cryptographic primitives used.

In computational models, often security is assessed by analyzing separately different classes of attacks. Distance-bounding protocols are not the exception, and there exist four main classes of attacks. Some authors consider more classes but, consistently with [CRSC12], our classification represents an exhaustive partition of the full space of attacks: *mafia fraud* [DGB88], *distance fraud* [Des88], *terrorist fraud* [DGB88] and *distance hijacking* [CRSC12]. We briefly describe next these attacks, and their graphical representations are depicted in Figure 1.4[2].

Mafia Fraud (Figure 1.4a) Given a verifier V, a close attacker E uses a distant and honest prover P to make V believe that P is close. The attack works in two sessions: one between E and P (the so-called *pre-ask* session) and another one between V and E. The attacker E typically relays the verbatim untimed communication between P and V, and impersonates P to V during the fast phase.

Distance Fraud (Figure 1.4b) Given a verifier V, a distant and dishonest prover E replies in advance to V's challenges so the V believes that E is close. This means that E must be able to reproduce the responses prior to receiving the challenges.

Terrorist Fraud (Figure 1.4c) Given a verifier V, a close and dishonest prover E and a distant prover E' *collude* (any deviation from their protocol specification) to make

[2]Thanks to https://thenounproject.com for the icons.

V believe that E' is close. A condition for this attack to be valid is that, without further collusion involving E', it must not be possible to convince V that E' is close in further sessions of the protocol.

Distance Hijacking (Figure 1.4d) Given a verifier V, a distant and dishonest prover E makes use of close and honest provers to make V believe that E is close. To achieve this, E lets the honest provers run the fast phase with V. Then (or sometimes before) E hijacks the session to inject their own identity-defining messages, possibly during the verification phase of the protocol. This tricks V into believing that they are running the protocol with E.

Two notes regarding the above descriptions of the attacks are as follows. First, some authors consider mafia fraud and relay attacks to be equivalent. We do not, because mafia fraud attackers can manipulate the messages rather than simply relaying the verbatim messages exchanged between the legitimate parties. Second, we note that E' must not be under full control of the adversary, otherwise the condition of no further false proximity proof does not hold. In particular, E' must not reveal their secret key material.

On Computational Analysis of Security

Distance-bounding protocols aim to minimize the processing time on the prover side during the fast phase. A small and deterministic processing time will improve the precision of the distance estimation and thus prevents attacks in which the adversary *overclocks* the prover [CHKM06, BC93, HK08]. To minimize such computational time, it is common to use inexpensive operations such as exclusive-OR or a lookup operation on a constant-time access table, built up prior to the fast phase.

A large number of distance-bounding protocols proposed to date indeed employ the lookup-based approach, e.g. [HK05, TP07, MP08, AT09, KA09, TMA10, GAA10, KKBD11, ALM11, TMA14] Another common principle that many protocols share is that the prover does not send any messages after the fast phase has been completed. This might provide some guarantees on proximity and authentication, even if the protocol execution is halted after a few RTT measurement rounds. We will focus on these protocols, i.e. those in which (1) the prover's fast phase messages are looked-up from a table built up prior to the fast phase, and (2) the prover does not send any messages after the fast phase. We call these protocols *lookup-based* distance-bounding protocols, or lookup-based protocols for short.

Lookup-based protocols typically consist of two phases: slow and fast. In the slow phase, the verifier and the prover agree on the parameters to precompute a lookup table with *all* possible responses for the fast phase. Then, at the i-th round of the fast phase, the verifier sends a random challenge bit c_i to the prover and starts a clock. The prover replies instantly to the challenge c_i by looking up the response r_i, according to c_i, from the lookup table. Upon reception of the prover's reply, the verifier stops the clock to determine the round-trip time Δt_i. The protocol completes correctly if all responses are correct and $\Delta t_i \leq \Delta t_{\mathsf{max}}$ for all rounds, where Δt_{max} is again a predefined threshold.

A well-known, possibly the earliest, lookup-based distance-bounding protocol is that of Hancke and Kuhn [HK05] (HK). The protocol is depicted in Figure 1.5 and works as follows. In the slow phase, the verifier V sends a random nonce N_V to the prover P, who creates

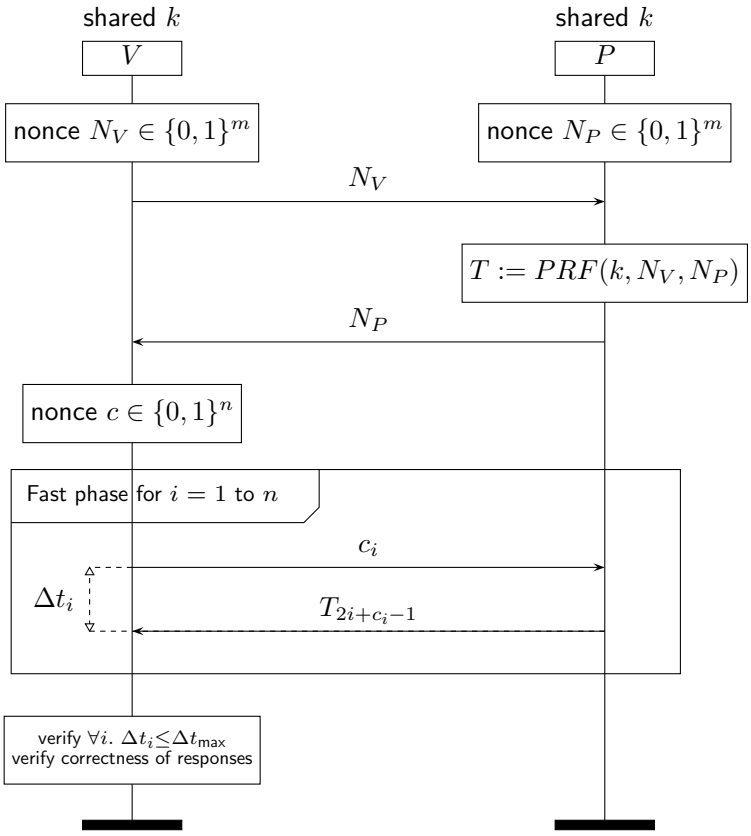


Figure 1.5 Hancke and Kuhn's (HK) protocol [HK05], where m is a publicly-known integer number and PRF is a public pseudo-random function whose output is a $2n$-bit string.

a bit sequence T from the output of a *pseudo-random* function PRF on the long-term symmetric key shared between V and P, the verifier's nonce N_V, and the prover's own nonce N_P. The prover then sends N_P back to V. Upon reception, V starts the fast phase, composed of n round-trip time measurement rounds numbered from 1 to n. At the i-th round, V sends a random challenge bit c_i to which P must reply with the bit T_{2i+c_i-1}. The protocol completes correctly if all responses are correct and the round-trip times are below the threshold.

The simplicity of lookup-based protocols makes them attractive for ubiquitous wireless technologies such as RFID and NFC. The apparent drawback of these protocols is that they fall short in terms of resistance to mafia fraud in comparison to more cryptographically expensive approaches, such as the BC protocol. To motivate this, let us consider again the HK protocol.

An adversary can execute a mafia fraud attack against the HK protocol as follows. First, the adversary relays all communication between the prover and the verifier during the slow phase. Before the start of the fast phase, an adversary E poses as the verifier V and queries the prover P with the same challenge 0 for each one of the n rounds. As a result, E receives the sequence of responses $T_1, T_3, \ldots, T_{2n-1}$ from P. Then, in the second session, the adversary uses this knowledge to reply to V's challenges during the fast phase. The probability of success of an adversary executing such an attack is $\left(\frac{3}{4}\right)^n$. This value is

obtained from the following reasoning. At the i-th round of the fast phase, E is challenged with a random bit c_i. If $c_i = 0$ then E answers with the correct reply T_{2i-1}. Otherwise, E replies with a random bit, which gives a chance of $\frac{1}{2}$ to reply correctly. In total, this attack succeeds with probability $\frac{1}{2} \times 1 + \frac{1}{2} \times \frac{1}{2} = \frac{3}{4}$ per round, leading to $\left(\frac{3}{4}\right)^n$ for n rounds. This value is significantly larger than the probability $\left(\frac{1}{2}\right)^n$ [BC93] of the same attack to succeed against the BC protocol, also for n rounds.

However, Hancke and Kuhn [HK05] explained the advantage of avoiding a final slow phase. Their reasoning is that the execution time of a few additional rounds during the fast phase is significantly lower than that of performing extra cryptographic operations. Hence, for all values of n, there exists u larger (but not much larger) than n such that the HK protocol with u rounds is more resistant to mafia fraud than the BC protocol with n rounds, i.e. $\left(\frac{3}{4}\right)^u < \left(\frac{1}{2}\right)^n$.

In 2009, Avoine and Tchamkerten proposed a lookup-based protocol whose lookup structure is an edge-labeled binary tree. This protocol reduces the probability of a mafia fraud attack to succeed to $\frac{1}{2^n}(1 + \frac{n}{2})$ [AT09], in comparison to $\left(\frac{3}{4}\right)^n$ of the HK protocol. However, this good performance of Avoine and Tchamkerten's protocol (Tree) comes at the price of an exponential space complexity in the number n of fast phase rounds. Indeed, the n-depth binary tree structure that their protocol employs gives it exponential space complexity. Space complexity is an important consideration in lookup-based protocols, and in distance-bounding protocols in general, as it determines the amount of memory a device running the protocol must be able to allocate. Memory is indeed a critical issue in ubiquitous technologies such as RFID.

It is not known if the security of the Tree protocol [AT09] can be achieved at lower space costs. Furthermore, it is also an open question whether this lower bound $\frac{1}{2^n}(1 + \frac{n}{2})$ can be reduced in lookup-based protocols. This discussion on the prominent class of lookup-based protocols, which all have structural similarity, leads to our first research question.

> **Research Question 1.** *Can we define a computational model that would allow us to comprehensively study security and other properties of lookup-based distance-bounding protocols?*

In order to develop a comprehensive and comparative analysis of lookup-based protocols, we propose a computational model that represents a protocol as a set of *state-labeled* Deterministic Finite Automata. An execution of the protocol is thus represented by a random walk in a randomly selected automaton. The adversary is modeled by any polynomial algorithm whose input is the walk in the selected automaton.

An extensive number of lookup-based distance-bounding protocols have been published to date. Each of them aims to bring improvement on its predecessors not only in terms of security, but also in space complexity, defined by the *size* of the protocol. Despite the efforts on optimizing these comparison criteria, no protocol has been proven optimal yet. More precisely, it still remains an open problem how to design a lookup-based protocol whose resistance to mafia fraud is optimal, given an upper bound on the size of the protocol. This leads to our second research question.

> **Research Question 2.** *Within a prominent class of lookup-based protocols, can we design a protocol that optimizes the trade-off between resistance to mafia fraud and the size of the protocol?*

By using the proposed automata-based computational model, we provide a formal definition of the security and size trade-off optimization problem in lookup-based protocols. Furthermore, we give a solution protocol for this problem, for a large subclass of lookup-based protocols.

On Symbolic Analysis of Security

The majority of academic work on distance-bounding protocols use a given computational model (e.g. [ABK+11, DFKO11, BV14]) to conduct security analysis of distance-bounding protocols. In general, this analysis is assessed in terms of provable-resistance to mafia and distance frauds. To compute the probability of success of terrorist fraud and distance hijacking attacks does not seem to be a common approach. In fact, resistance to these two types of attacks has been traditionally given in a "yes or no" fashion (an attack assertively exists or not) rather than in a probabilistic way.

A few symbolic frameworks for analysis of distance-bounding security exist: [BCSS09, SSBC09], [CRSC12], [MSTT18b], [CdRS18], [DDW18] and [MSTT19]. The publications [MSTT18b, MSTT19] are part of this thesis.

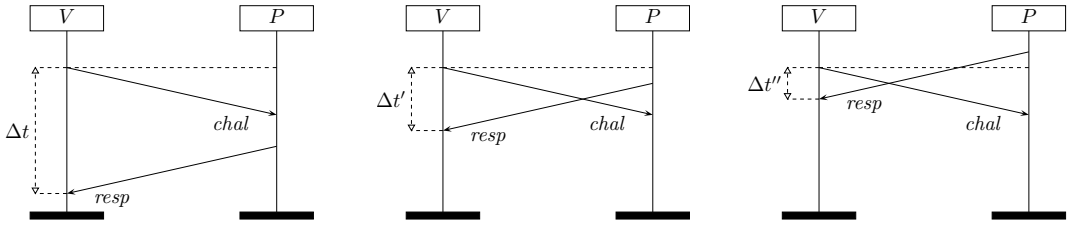
In 2009, Basin et al. proposed the first tool-supported framework [BCSS09, SSBC09] to verify distance-bounding protocols by employing a symbolic approach. This work constituted a major step forward, as previous (computational) security/attack proofs were complex and error prone, whilst symbolic proofs are simpler and often produced and verified by a computer software. Not long after Basin et al.'s proposed their framework, it proved further significance when Cremers et al. [CRSC12] introduced a new type of attack: distance hijacking, and found a number of protocols vulnerable to it. Surprisingly, many of those protocols had been proven secure in computational approaches. An important observation is that Cremers et al. assumed a Dolev-Yao [DY83] adversary, so they did not introduce a stronger adversary model to demonstrate their attack.

Unfortunately, while the desired security properties of a distance-bounding protocol can be precisely specified in Basin et al.'s framework, it is not so straightforward to verify them for a given protocol. This is because the verification relies on modeling *continuous time* and *location of agents* to deliver the security proofs by using an adapted version of the higher-order theorem-proving tool Isabelle/HOL [NPW02]. This tool is acknowledged to require a considerable amount of user intervention and expertise.

By comparison, well-established tools for automatic verification of security protocols, such as Tamarin [MSCB13], ProVerif [Bla01] and Scyther [Cre08], are able to verify traditional security properties in a straightforward and rapid way. These tools handle time as a discrete ordering of events, therefore specification of protocols featuring the notion of continuous time is unfeasible in these tools. This leads to our third research question.

> **Research Question 3.** *Can we define a property, equivalent to that of Basin et al.* [BCSS09, SSBC09] *for symbolic verification of distance-bounding protocols, that does not explicitly consider time or location?*

We argue that the notions of time and location are indeed *not* needed to specify and verify distance-bounding protocols. Surprisingly enough, such protocols can be verified by considering the causal order of events in protocol traces, similar to verification of



(a) Correct timing **(b)** False timing **(c)** "Very" false timing

Figure 1.6 Three timing scenarios of a challenge/response round.

authentication properties such as aliveness [Low97, CM12]. The intuition behind this observation is illustrated in Figure 1.6.

Figure 1.6a shows a regular challenge/response round, in which prover P can only respond to verifier V's challenge after having received the challenge. Therefore, $\frac{c}{2} \cdot \Delta t$ determines an upper bound on the distance between V and P. Suppose now that, due to a vulnerability of the protocol, P is able to predict the appropriate response before having received the challenge (Figure 1.6b). This means that he will be able to send his response "too early", leading to a shorter round-trip time $\Delta t' < \Delta t$ and thus to a smaller and incorrect distance calculated by V. Thus, if the protocol is insecure because P can preempt the response, P has sufficient knowledge to create the response before reception of the challenge. Hence, our main observation is that, assuming no other causal relation between sending the challenge and P's knowledge, P could even have sent the response *before V sent the challenge* (Figure 1.6c). From a causal point of view, this means that if there is a trace in which P sends its response before P receives the challenge, there must also be a trace in which P sends the response before V sends the challenge. Hence, a flaw in the protocol translates into a wrongly ordered trace, which can be discovered through an analysis that does not consider time.

Basin et al.'s formalism [BCSS09, SSBC09], and ours by extension, employs a Dolev-Yao [DY83] adversary model with a few restrictions. Such restrictions are inherent from the distance-bounding setting and are based precisely on the physical law that sets the speed of light as an upper bound on the ratio travel-distance over travel-time of a message. From a symbolic point of view, this law enforces that adversary data injection must be done through a compromised agent. The compromise capabilities of the adversary are the same as in the Dolev-Yao adversary, who compromises agents by exerting full control over them for the entire protocol execution after the compromise. Hence this formalism, and again ours either, do not capture terrorist fraud. We motivate this next.

Terrorist fraud is an attack in which provers *collude* in order to provide a verifier with a false proximity proof. Collusion is a form of secret aid given by agents who are not compromised by the adversary. Indeed, collusion differs from compromise in that compromise is exerted by the adversary and collusion is a deliberate choice of the agent(s) involved. For example, *covert adversaries* [AL10, FY92, CO99] are agents who are willing to cheat by deviating from the protocol specification, as long as the cheating would not be detected. One might think of an online gaming platform, in which some players secretly cooperate to cheat against other players, whilst avoiding being caught, or else face consequences such as being thrown out of the platform.

In order to reason about terrorist fraud within our formalism, we must account for collusion in security protocols. Collusion can be modeled by allowing non-compromised agents to deviate from their protocol specification. This leads to the fourth and last research question of this thesis.

> **Research Question 4.** *Can we build a tool-supported framework for symbolic verification of distance-bounding protocols that accounts for all four classes of attack?*

Two main approaches to defining terrorist fraud attacks exist. Both state that a distant prover, by colluding with a close and compromised prover, make a verifier believe that the distant prover is close. The approaches differ in the condition on the collusion that make one consider the attack as valid. The first approach states that a terrorist fraud occurs whenever the distant prover does not reveal their secret keys in the process of collusion. The second approach considers a weaker condition, in which the distant prover must not allow the compromised prover to proof proximity in further sessions without further collusion. In this thesis we adopt the second approach, which we believe it to be more accurate than the first approach. We briefly discuss the reasons in the next two paragraphs.

It is significantly hard to properly model the meaning of "the distant prover does not reveal their secret keys in the process of collusion". This is because key reveal must be explicitly modeled in most (if not all) automated verification tools as part of the protocol/environment specification, and is always the case that some key reveal scenarios are missed in this modeling.

The second reason is also related to the reveal of secret keys or rather to the reveal of messages that are as relevant as secret keys. For example, for some protocols, an agent's leakage of session-fresh data can lead to their impersonation in every session thereafter. A running example is given as follows. Consider a distance-bounding protocol *Proto* in which every crypto-operation uses a shared, symmetric key. Let us assume that the only aid the distant prover can provide the close prover with is to give away the shared key. If we follow the first approach, *Proto* would be resistant to terrorist fraud. Consider now another protocol *Proto*$'$ that results from *Proto* by replacing any instance of a shared key k by its hash $h(k)$. Therefore, if we follow the first approach, *Proto*$'$ would not be resistant to terrorist fraud, as the distant prover can leak $h(k)$, which does not reveal k. This means that the key-hashing transformation weakens the protocol, which does not seem to be a coherent statement. The mentioned issue does not occur if we follow the second approach as both *Proto* and *Proto*$'$ would be resistant to terrorist fraud.

By providing a formal definition of (resistance to) terrorist fraud, we develop a symbolic verification framework[3] that accounts for the four classes of attacks on distance-bounding protocols. The framework is developed in Tamarin and is highly modular, thus users only have to specify the protocol and the framework will generate and verify the concerning generic security lemmas.

[3] Available at https://github.com/jorgetp/dbverify

1.3 Contributions

The four main contributions of this thesis are summarized as follows:

(1) We propose an abstract model to study theoretical properties for a large class of distance-bounding protocols, called *lookup-based* protocols. The model represents a protocol as a set of Deterministic Finite Automata (DFA), and protocol executions are represented as random walks in random automata of the set. We prove that $\frac{1}{2^n}(1+\frac{n}{2})$ is a tight lower bound on the resistance of lookup-based protocols to mafia fraud, where n is the number of fast phase rounds. We define a pre-ask strategy to execute a mafia fraud attack that is optimal in the prominent class of *layered and random-labeled* lookup-based protocols. The strategy is to reply to the verifier's challenges with exactly the prover's responses from the pre-ask session. The Tree protocol [AT09] achieves the mentioned optimal resistance to mafia fraud, but it is unfeasible in practice due to its exponential space complexity. We thus define a family of protocols which strike excellent resistance to mafia fraud in relation to their space complexity.

(2) By using the proposed model based on DFAs, we study trade-offs between security and space complexity, in layered protocols. We define two equivalence relations on DFAs and use them to demonstrate our optimality result: given an upper bound on the protocol size (the size of a protocol is the number of states of the largest layer amongst all automata of the protocol), the protocol whose automata have the largest girth (the girth of an automaton is the shortest cycle if viewed as an undirected graph) is *optimally* resistant to mafia fraud amongst all layered and random-labeled protocols. We provide a concrete construction of a protocol whose automata have the largest girth, which we call the *Modular* protocol. We use the multi-criteria comparison framework [AMT15] to show that our protocol compares well with protocols that are non-layered or even not lookup-based.

(3) We build on Basin et al.'s symbolic model [BCSS09, SSBC09] in order to define a *causality*-based property to verify distance-bounding protocols. Our property does *not* consider continuous time or agents' location. Instead, it resembles a form of *aliveness* [Low97, CM12] in that the prover must perform some action during the fast phase of the protocol. Specifically, we demonstrate that a verifier's guarantee that the prover (or any compromised agent if the prover is compromised) is alive during the fast phase is equivalent to the verifier's guarantee that the fast phase round-trip time can be used to upper-bound their distance to said prover.

(4) We develop a multiset rewriting formalism to reason about *collusion* in security protocols. Collusion is any deviation from the protocol specification of agents who are not under full control of the adversary. By relying on this formalism, we extend our distance-bounding security model in order to account for *terrorist fraud* attacks. By using our extended model, we develop a Tamarin-based symbolic verification framework that accounts for all four classes of attack from the literature on distance bounding. We also conduct a security survey of over 25 protocols, which include industrial protocols such as Mastercard's contactless payment protocol PayPass and NXP's MIFARE Plus with proximity check. For the industrial protocols we confirm known attacks, propose fixes, and provide Tamarin-constructed proofs of security of the repaired protocols.

1.4 Overview

This thesis consists of two parts and four research questions. Each part addresses two research questions. The first part concerns security analysis of distance-bounding protocols in the computational model. This part is based on the publications [MTT16a, MTT16b], which are joint works with Sjouke Mauw and Rolando Trujillo Rasúa. The second part develops a symbolic framework for security analysis of distance-bounding protocols. This part is based on the papers [MSTT18b, MSTT19], which are joint works with the aforementioned co-authors and Zach Smith. The author of this thesis also co-wrote the papers [STBF17, MSTT18a] while developing this PhD. These papers are not particularly within the scope of this dissertation, thus not included in it.

Part I: Computational Analysis of Distance-Bounding Protocols

Chapter 3 addresses Research Question 1. This chapter focuses on the class of lookup-based distance-bounding protocols. By designing an automata-based model for these protocols, we comprehensively study their properties, such as security lower bounds in relation to space complexity. Further, we develop a novel family of protocols within this class that resist well to mafia fraud. This chapter is based on the paper:

- Sjouke Mauw, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. A class of precomputation-based distance-bounding protocols. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pages 97–111, 2016 [MTT16a].

Chapter 4 addresses Research Question 2. In this chapter we study the trade-off between security and size for a non-trivial class of lookup-based protocols. As a result, we construct a protocol that strikes the optimal resistance to mafia fraud within such class, for a given upper bound on the size of the protocols. By using a tool-supported multi-criteria comparison method [AMT15], we show that the protocol compares well with protocols outside the analyzed class. This chapter is partially based on the paper:

- Sjouke Mauw, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Optimality results on the security of lookup-based protocols. In *Radio Frequency Identification and IoT Security - 12th International Workshop, RFIDSec 2016, Hong Kong, China, November 30 - December 2, 2016, Revised Selected Papers*, pages 137–150, 2016 [MTT16b].

Part II: Symbolic Analysis of Distance-Bounding Protocols

Chapter 5 addresses Research Question 3. We first describe the symbolic framework by Basin et al. [BCSS09, SSBC09], which models timestamp of agents' events and agents' location. We build on Basin et al.'s model to provide a causality-based characterization of secure distance-bounding. This chapter is based on the paper:

- Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *2018 IEEE Symposium on Security and Privacy, S&P 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 549–566, 2018 [MSTT18b].

Chapter 6 partially addresses Research Question 4. We first develop a multiset rewriting formalism to reason about collusion (a deviation from the protocol specification) in security protocols, and introduce the notion of post-collusion security, which verifies security properties claimed in sessions initiated after the collusion. By means of post-collusion security, we extend the distance-bounding security model of Chapter 5 in order to account for terrorist fraud. This chapter is based on the paper:

- Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Post-collusion security and terrorist fraud. 2019. (under submission) [MSTT19].

Chapter 7 partially addresses Research Question 4. By using the definitions from Chapters 5 and 6, we develop a TAMARIN framework for symbolic verification of distance-bounding protocols. With our framework, we perform an extensive survey of distance-bounding protocols from both academia and the industry. The protocol models and proofs can be freely accessed online[4]. This chapter is based on the paper:

- Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Post-collusion security and terrorist fraud. 2019. (under submission) [MSTT19].

[4] At https://github.com/jorgetp/dbverify

2

Related Work

Possibly more than forty distance-bounding protocols have been proposed to date. Most of them can be classified in one of two major classes: lookup-based protocols (which we briefly introduced in Chapter 1 and for which have two dedicated chapters in this thesis), and protocols based on Brands and Chaum's [BC93] commit/authentication approach. The most significant difference between these classes is the presence or not of a final, prover-active, authentication phase. We do not intend to list distance-bounding protocols here. Instead, we will describe previous works that develop frameworks for distance-bounding security analysis within computational and symbolic models. We will also give our own views on the described approaches and their results.

2.1 On Computational Analysis

Almost every protocol published to date comes with a specific, isolated computational approach for proving their security. This section, however, focuses on various of those works in which security analysis is conducted in a systematic and generic manner, thus making them possibly applicable to further protocols.

Avoine, Bingöl et al.'s Framework

Avoine, Bingöl et al. [ABK+11] introduced a unified security analysis framework that includes a thorough terminology about the frauds, adversary, and prover. The authors explore the adversary's capabilities and strategies, and the effect of the provers' abilities to tamper with their devices. To develop their security analyses, they focus on four types of frauds: impersonation, distance, mafia and terrorist frauds. The particular case of impersonation regards a lonely prover who pretends to be another one.

This work considers three adversarial strategies to execute the attacks: *pre-ask*, *post-ask*, and *early-reply*. In the pre-ask strategy (which we briefly discussed in Chapter 1 and will formalize in Chapter 3), the adversary relays the slow phase messages between the verifier and the prover, runs a fast phase with the prover, and then runs (on behalf of the prover) a fast phase with the verifier. In the post-ask strategy, the adversary relays the slow phase messages, executes the fast phase with the verifier without involving the prover, and then queries the prover with the correct challenges received during the fast phase with the verifier. The authors suggest that post-ask strategy is meaningful when the protocol

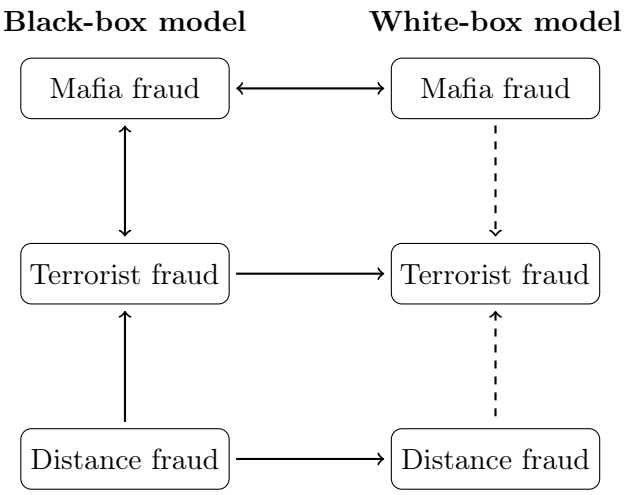


Figure 2.1 Relations between the frauds in the white-box and black-box models as given in [ABK+11]. An arrow (dashed or not) from A to B means that for any attack in A that succeeds with probability p_A, an attack in B exists that succeeds with probability $p_B \geq p_A$. Dashed arrows are relations we do not subscribe to.

features a final phase that checks that the challenges received by the prover are correct. Finally, the early-reply strategy refers to an adversary who anticipates the replies to the verifier's challenges so the replies arrive earlier than they should if sent upon reception of the challenges. This strategy is particularly relevant for distance fraud attacks.

The authors also introduced new concepts in the distance-bounding field, such as *black-box* [Bla00, YY96] and *white-box* [CEJvO02, SWP09] models. On the one hand, in their black-box model, the prover cannot observe or tamper with the execution of the algorithm. In their white-box model, on the other hand, the prover has full access to the implementation of the algorithm and a complete control over the execution environment. This works also provides relations between the frauds when considering both white-box and black-box models. The relations are illustrated in Figure 2.1. We do not agree with (at least) two of such relations, which are the ones represented with dashed arrows in Figure 2.1. For example, the SPADE protocol [BGG+16] is vulnerable to mafia fraud (with probability 1) in the white-box model, yet it is resistant to terrorist fraud. Furthermore, any protocol in which the fast phase responses are constant is vulnerable to distance fraud (and also to mafia fraud) but is terrorist fraud resistant. We do not have concrete protocol examples to refute any of the remaining relations.

Finally, the authors make use of their framework to provide a detailed analysis of Munilla and Peinado's protocol [MP08]. An outcome of their analysis is a higher probability of success of mafia fraud than the one due to Munilla and Peinado given in [MP08].

Dürholz et al.'s Formal Approach

Dürholz et al. [DFKO11] developed rigorous cryptographic security models for mafia, terrorist, and distance fraud attacks. The authors define their communication/attacker model such that honest parties reply as soon as they have received a (protocol) message, and the adversary schedules message delivery to honest parties. Their model also considers

Impersonation	Mafia Fraud	Distance Fraud	Terrorist Fraud
✓	✓	✓	×
✓	✓	×	✓
✓	×	✓	✓

Table 2.1 Relations between fraud types as per Dürholz et al. [DFKO11]. Each row of the table means that a distance-bounding protocol exists such that, for each fraud given by the columns, it is either secure (✓) or an attack exists (×).

a global clock that incrementally assigns (for every k and every sid) an integer number $\mathsf{clock}(sid, k)$[1] to the k-th protocol message delivered to an honest party, in a session with identifier sid. The model allows them to make claims on whether the adversary has tainted a particular fast phase or not. Interestingly, their notion of "activeness" of the adversary during the fast phase somehow relates to one of the main results of this thesis (developed in Chapter 5) in that we consider such activeness (which we call aliveness) but of the prover rather than the adversary, in order to deliver security proofs. Unfortunately, we were not able to draw more concrete connections between the two approaches.

Besides the three aforementioned frauds (mafia, terrorist, and distance), Dürholz et al. also consider impersonation attacks (a concept similar to that of [ABK+11]). They suggest a simple, yet strong definition of impersonation resistance as a basic requirement of identification. Whereas the three previous frauds concern the fast phase, impersonation security requires that an adversary cannot impersonate a tag in either the slow or the final phase. One could notice some degree of similarity between Dürholz et al.'s impersonation with distance hijacking (yet unknown at the time). Thus, it would be of interest to see the outcome of the framework, in terms of impersonation, when analyzing protocols that are vulnerable to distance hijacking attacks, such as Brands and Chaum's protocol [BC93].

Dürholz et al. also refute the claim in [RNTS07] that terrorist fraud resistance implies distance fraud resistance, which coincides with one of our claims related to the fraud relations of [ABK+11] illustrated in Figure 2.1. More generally, Dürholz et al. claim that the fraud types are rather independent, and demonstrate the existence of protocols that negate dependencies between the frauds (see Table 2.1).

The authors also study in detail Kim and Avoine's protocol [KA09], which is mafia fraud and distance fraud resistant, but it fails to defend against impersonation and terrorist fraud, according to [DFKO11]. Thus, Dürholz et al. amended this protocol so their version provides impersonation security and also deals with noisy channels. Furthermore, they provide formal proofs of security of their enhanced version of Kim and Avoine's protocol, in terms of impersonation, mafia and distance frauds. Dürholz et al. show that a terrorist fraud attack on their protocol version does exist though. We confirm their results in Chapter 7.

Avoine, Mauw and Trujillo's Multi-Criteria Comparison

Avoine, Mauw and Trujillo proposed in [AMT15] a tool-supported methodology to compare distance-bounding protocols. This work uses multi-citeria decision-making techniques to

[1] Some form of discrete timestamps.

deliver classes of *non-dominated* protocol instances. Protocols instances result from of various choices of protocol parameters. A protocol instance dominates another protocol instance if two conditions hold: (1) the former is at least as good as the latter in all criteria, and (2) the former is better than the latter in at least one criterion. A protocol instance is non-dominated if no other protocol instance dominates it.

The authors consider eight comparison criteria that have been used frequently in the literature: number of bits exchanged during the fast phase, probability of success of mafia, distance and terrorist fraud attacks, number of bits of the fast phase communication channel, number of cryptographic operations, memory, and whether the protocol has a prover-active final phase or not. Their comparison method constructs classes of winner protocol instances according to non-dominance Pareto frontiers. The comparison software[2] is written in Java and includes around thirteen fully-detailed protocols. Due to the modularity of the software source, new protocols can be easily added. We will use this framework in Chapter 4 (specifically Section 4.4) of this thesis.

Yang et al.'s Two-Hop Distance Bounding

The work by Yang et al. [YPM+18] extends traditional distance-bounding protocols to a two-hop setting. In this setting, the prover is out of the verifier's communication range, hence they must rely on an untrusted in-between entity in order to verify proximity. This work presents a formal framework to compute the probability of success of mafia, distance and terrorist frauds against *register-based* distance-bounding protocols (a subclass of what we call layered lookup-based protocols, which we briefly introduced in Chapter 1 and will formally study in Chapters 3 and 4). The authors also provide a general method to extend traditional distance-bounding protocols to the two-hop case.

The authors developed their framework by considering a well-defined set of functions that model the protocol specification. They determined the success probability of the aforementioned three types of attack (mafia, distance, and terrorist frauds) against several state-of-the-art protocols, and conducted simulated experiments in order to validate their theoretical security analyses. They also showed that the presence or not of a third, linker entity in a two-hop protocol affects the security in relation to its corresponding one-hop protocol, as opposed to what it was believed before.

While the class of register-based distance-bounding protocols indeed covers a large number of protocols published to date, it does not seem hard to escape from their round-independence scheme by re-defining the prover's fast phase function domain so it considers not only the current round's challenge but also every challenge the prover has received before the current round. This generalization obviously affects the probabilistic security analysis but it could lead to perhaps more interesting results, applicable to a larger class of protocols.

Avoine et al.'s Survey

Avoine et al. [ABB+19] developed a detailed survey of eleven distance-bounding protocols with their variations. The survey assesses security against impersonation, distance fraud, mafia fraud, and terrorist fraud attacks; and they focus on the three adversary strategies

[2] Available at https://github.com/rolandotr/db_comparison

pre-ask, post-ask and early-reply. The survey also analyzes performance aspects of the protocols such as: cryptographic primitives (type of cryptographic primitives needed to be implemented on the prover side), number of exchanged bits during the slow phase, number of exchanged bits during the fast phase, and memory consumption. One must notice the similarities between this work and the works [ABK+11, AMT15].

2.2 On Symbolic Analysis

We will discuss on various works on symbolic verification of distance-bounding protocols. To the best of our knowledge, the works revised in this section, along with our papers [MSTT18b, MSTT19] (which are part of this dissertation), form an exhaustive list of computer-assisted frameworks for distance-bounding symbolic verification. For the sake of simplicity, the protocols illustrated in this section abstract away from cycles of RTT measurement rounds into a single one.

Meadows et al.'s Authentication Logic

Meadows et al. [MPP+07] proposed a symbolic framework based on the combined authentication and secrecy logics of [CMP05, PM06]. Their logic defines two basic axioms: the *receive* axiom for reception of messages and the *freshness* axiom for modeling freshness of terms (or messages). The receive axiom states that every message which is received must have been originated from someone. The freshness axiom states two properties: (1) any event mentioning a fresh message must occur after such message has been generated, and (2) every fresh message that is known to an agent A and was not generated by A must have been received (ruled by the receive axiom) from another agent.

Meadows et al.'s formalism also considers further axioms to deal with authentication codes (which refer to the deduction of the agent who created a particular message), and events' timestamps and distances between agents. Their approach is particularly oriented to security analysis in the presence of honest provers.

In the case of dishonest provers, Meadows et al.'s methods are not able to derive meaningful security results. The reason is that, as the authors claim, they would require strong assumptions about the behavior of other dishonest agents well. For example, suppose a dishonest prover P sends out a predictable value instead of a fresh nonce as meant by the protocol specification. Then a dishonest prover E, who is closer to the verifier than P is, could have anticipated and sent P's rapid response, thus tricking the verifier into believing that P is close. In order to account for this kind of scenarios, the authors claim they would need to make the assumption that E could not produce P's response in advance. The authors point out that this assumption does not differ from considering honest prover behavior. The authors also make a connection between this scenario and terrorist fraud.

Meadows et al. also proposed a novel distance-bounding protocol, illustrated in Figure 2.2. Noticeable novelties of their protocol in relation to then-traditional distance-bounding protocols are twofold: (1) its fast phase uses a multiple-bit communication channel, and (2) the prover includes their location in the final authentication message.

The authors provide a formal security proof of their proposed protocol, supported by their logic. However, Cremers et al. indicated in [CRSC12] that Meadows et al.'s protocol

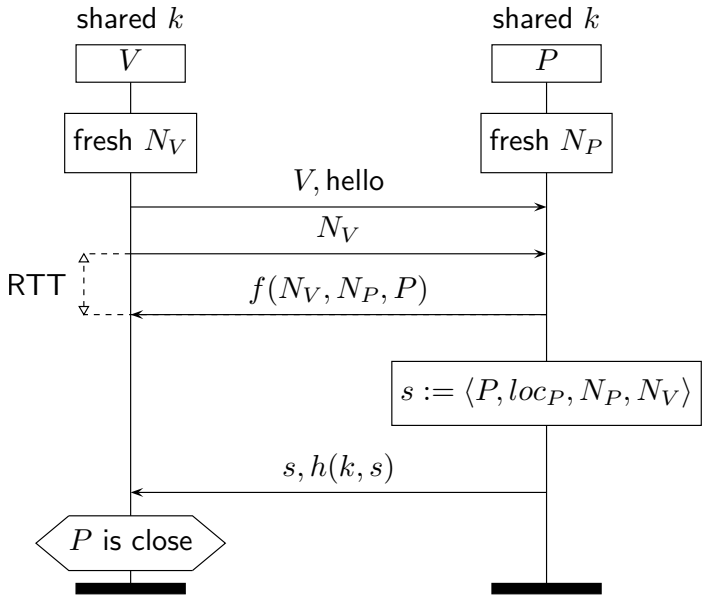


Figure 2.2 Meadows et al.'s protocol [MPP+07], where loc_P denotes the location of the prover P. Three instances of the protocol are given by the authors, as per the following three choices $f := \langle N_V, N_P \oplus P \rangle$, $f := \langle N_V, P, N_P \rangle$, and $f := N_V \oplus h(P, N_P)$ where h is a collision-free hash function. The symbol $\oplus$ represents the exclusive-OR operator.

with the choice $f := \langle N_V, N_P \oplus P \rangle$ is vulnerable to a distance hijacking attack (recall that in this attack, a distant and dishonest prover convinces the verifier to be close by using a close and honest prover). The attack is shown in Figure 2.3, which is indeed consistent with the authors' own claim that they do not cover attacks of the dishonest-prover type. In Chapter 7 we will confirm this vulnerability in their protocol.

Finally, Meadows et al. briefly address collusion analysis in distance bounding. The desired outcome of collusion is to make a verifier believe a collusion-involved prover is closer than they are. The authors illustrate this by using *standard* collusion and also *wormholes*. Their standard collusion refers to a pool of shared information (which may include long-term secrets) from which any colluding agent can extract information. Their wormhole attack, on the other hand, is one in which a fast channel is set up between the victims and an distant attacker. No formal analysis related to collusion is provided. In addition, the authors fail to make a connection between collusion and terrorist fraud.

Malladi et al.'s Automatic Analysis

The (possibly) first formal framework for distance-bounding protocols with multi-prover scenarios was proposed by Malladi et al. [MBK10], along with a software tool. They analyzed the signature-based Brands and Chaum's protocol [BC93] and found an attack in which an adversary who is not in the vicinity of the verifier still passes the protocol (a form of distance hijacking attack). They called this attack the *farther adversary* scenario. Moreover, to solve the security issue they found, they observed that including the prover's identity in the signature would make the protocol no longer vulnerable to farther adversary attacks. Their *constraint solver* tool only considers a bounded number of protocol processes

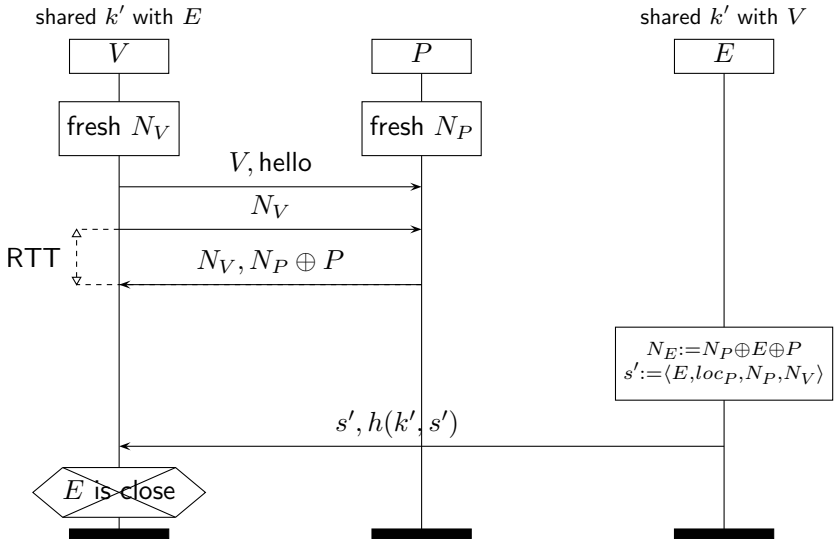


Figure 2.3 A distance hijacking attack on Meadows et al.'s protocol with $f := \langle N_V, P \oplus N_P \rangle$. In this attack, the legitimate prover P is close to the verifier V, and the dishonest prover E is far from V. The dishonest prover E hijacks the session by replacing P's final authentication message with their own authentication message, thus making V believe that E is close. E learns all messages exchanged between V and P by observation. The attack works because $N_E \oplus E = (N_P \oplus E \oplus P) \oplus E = N_P \oplus P$.

but the authors claim that their method can be easily extended to unbounded verification tools such as ProVerif [Bla01]. Though, we were not able to follow-up on such extension.

Basin et al.'s Isabelle/HOL-Based Framework

Basin et al. [BCSS09, SSBC09] presented a formal model that extends inductive and trace-based reasoning to include physical proximity in security protocols analysis. The authors refine the standard Dolev-Yao model to account for network topology, transmission delays, and agents' physical locations. This resulted in a distributed intruder with restricted, yet more realistic, communication capabilities. Their network model establishes that an (honest or dishonest) agent B may only receive, at time t, a given message sent at time t' by an (honest or dishonest) agent B if the following inequality holds:

$$t \geq t' + \frac{dist(A, B)}{\mathsf{c}}$$

where c denotes the network transmission speed and $dist(A, B)$ is the physical distance between A and B. This restricts the adversary's control of the network in two ways: (1) the adversary's injection of messages must be done via a dishonest agent with an actual physical location, and (2) there are delays between sending and receiving messages. This modeling allows one to verify validity of claims about proximity of agents.

Furthermore, the authors developed an abstract message theory that formalizes protocol-independent facts about messages. The authors formalized their model in the theorem-proving assistant Isabelle/HOL [NPW02] and used it to verify or provide attacks on a few

distance-bounding protocols[3]. We build on this work to develop our symbolic distance-bounding security model in Chapter 5, hence more details on this work will be given then.

Chothia et al.'s Hierarchy and ProVerif-Based Framework

Chothia et al. [CdRS18] presented an extension of the applied pi-calculus that can be used to model distance-bounding protocols. This work identified a new attack, which they call *uncompromised distance bounding attack*, in which the prover being tested is not compromised though other provers may have been. Their property however resembles a form of mafia fraud, and we do not find the differentiation meaningful.

This work seems to assume a difference between the attacker and a compromised (or dishonest) prover. That is, processes modeling dishonest provers are different from the process modeling the attacker. We argue that such differentiation does not seem to derive meaningful conclusions in distance-bounding analysis, except perhaps for odd cases in which devices have trusted hardware, which translates into the attacker not being able to compromise any prover. Our argument is supported by the fact that any action of the attacker is done through a compromised prover, therefore no different claim can be made about the "attacker's location" or a compromised prover's location.

The authors show how a number of different attacks can be encoded in their model and proved a partial order (or hierarchy) between them (see Figure 2.4). A summary of their discussion on the implications of their hierarchy of distance-bounding attacks is as follows:

(1) If the attacker model does consider terrorist fraud provers, then the strongest attack that needs to be defended against is assisted distance fraud.

(2) If the attacker model does not consider terrorist fraud provers, then strongest protection the protocol needs is against both distance hijacking and uncompromised distance-bounding attacks.

(3) If the attacker model does not consider compromised provers, then the strongest attack that needs to be defended against is uncompromised distance-bounding attack.

(4) If the attacker model assumes only trusted hardware devices, then the strongest attack that needs to be defended against is relay hijacking.

(5) If the attacker model only considers attackers that are distant to the verifier, then the strongest attack that needs to be defended against is distance hijacking.

The first and second implications are derived from the analysis of the red line of the figure. The third, fourth, and fifth implications are deduced from the analysis of the yellow, green, and purple lines of the figure, respectively.

Chothia et al. also showed how to compile their new calculus into the applied pi-calculus so that protocols can be automatically checked with the ProVerif tool [Bla01]. They used their framework[4] to analyze industrial protocols from Mastercard and NXP.

[3] Isabelle/HOL models and proofs available at http://www.infsec.ethz.ch/research/software/protoveriphy.html

[4] Source codes at http://www.cs.bham.ac.uk/~tpc/distance-bounding-protocols/CheckDB-master.tar.gz

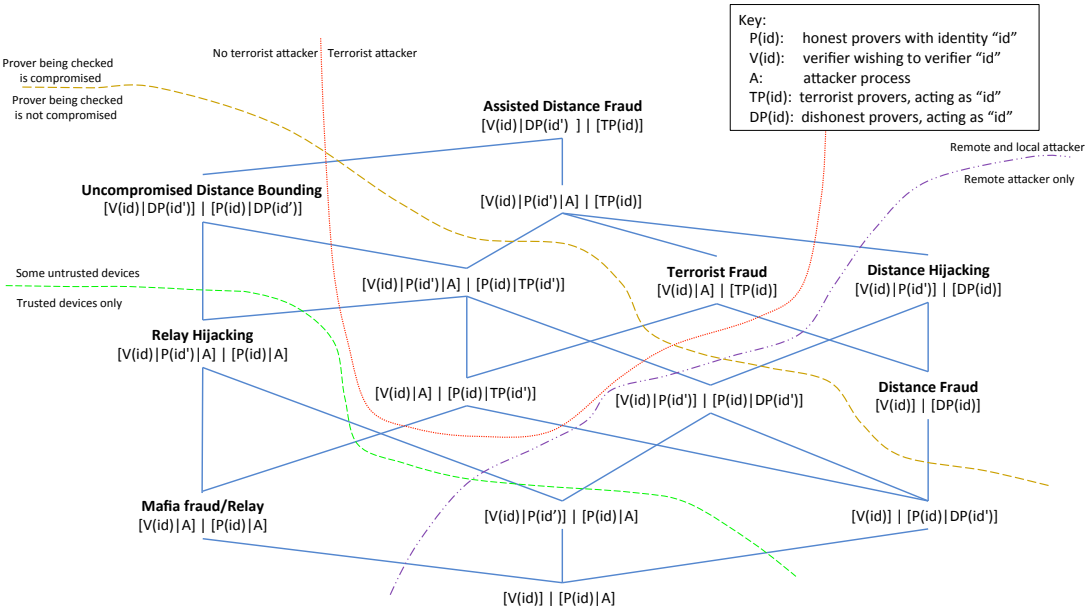


Figure 2.4 The hierarchy of distance-bounding attacks given by Chothia et al. [CdRS18]. The existence of a higher attack implies that any attack below it exists as well. The notation $[P_1|\ldots|P_n] \mid [Q_1|\ldots|Q_m]$ indicates that the processes P_i are co-located, the processes Q_i are co-located as well, and no process P_i is co-located with a process Q_j. This last statement implies that no timed communication occurs between P_i and Q_j.

To the best of our knowledge, this work brings the first and only published symbolic formalization of terrorist fraud. Recall that in terrorist fraud attacks, provers *collude* to falsely provide proximity proofs to a verifier. Collusion refers to actions, of agents who are not under control of the adversary, that are not specified by the protocol. Chothia et al. model collusion actions in the form of cryptographic oracles. A cryptographic oracle outputs the result of a cryptographic operation (such as encryption, decryption, signing, keyed-MACing) on an adversary-chosen input, without revealing the long-term keys.

In Chapter 7 (specifically Section 7.2) we will argue that their terrorist fraud modeling is not sound, provide our own modeling, and discuss differences between Chothia et al.'s approach and ours, not only for terrorist fraud but also for other frauds. Specifically, we will show that Chothia et al.'s framework delivers incorrect results for some protocols.

Debant et al.'s ProVerif-Based Framework

To model timed protocols and physical proximity, Debant et al. [DDW18] proposed a calculus that subjects communication to physical restrictions. The restrictions are applied to honest and dishonest participants (attacker). Similar to Basin et al.'s [BCSS09, SSBC09], an attacker can only send and receive messages in accordance with the sender and receiver's physical location. This implies that dishonest participants cannot instantaneously share or exchange knowledge. Instead, they must transmit the messages they want to share and therefore those messages take time to become known to the receiver(s).

Debant et al. [DDW18] provided reduction results motivated by [CC03] for traditional

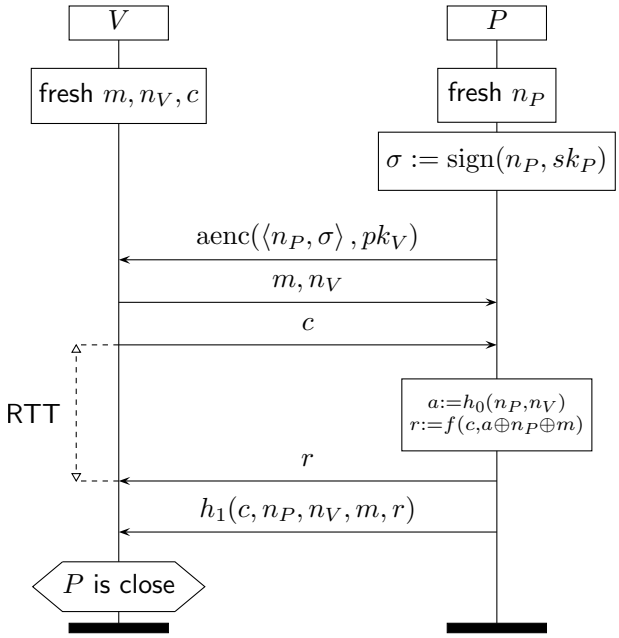


Figure 2.5 The SPADE protocol [BGG+16], where pk_X and sk_X respectively denote the long-term public and secret key associated to the agent X, $\text{aenc}(m, pk_X)$ denotes the asymmetric encryption of m with the (asymmetric) key pk_X, and $\text{sign}(m, sk_X)$ denotes the signature of m by X. A signature $\text{sign}(m, sk_X)$ is verified upon knowledge of m and pk_X.

protocols: if there is an attack, then there is one considering only few participants at some specific locations. Their work considers two classes of attacks: mafia fraud and distance hijacking. Their results slightly differ depending on the type of attack considered and allow one to reduce the number of topologies to be considered from infinitely many to only one (involving at most four participants including the malicious ones). Their reduction results hold for arbitrary cryptographic primitives as soon as they can be expressed using rewriting rules modulo an equational theory.

Their reduction results allowed them to use traditional verification methods and tools, such as ProVerif [Bla01]. They used this tool to analyze various distance-bounding protocols. Their case studies include the PaySafe protocol [CGdR+15], which is a distance-bounding version of Visa's payWave contactless payment protocol [EMV18b]. Their verification[5]. confirmed a number of known results, and discovered an unreported mafia fraud attack on the SPADE protocol [BGG+16]. Figures 2.5 and 2.6 illustrate the SPADE protocol and Debant et al.'s mafia fraud attack against it, respectively.

Debant et al. also analyzed the PaySafe protocol [CGdR+15], which is a distance-bounding-enabled version of Visa's payWave [EMV18b]. Their verification reports this protocol as secure against mafia fraud, and therefore against relay attacks as well, which is indeed the fundamental security goal of the protocol. In line with [CdRS18], Debant et al. do not consider dishonest-prover type of attacks to be relevant for payment protocols. We do not fully agree with this and will briefly argue about it in Chapter 7 (specifically Section 7.3). Furthermore, we will show that this protocol, and other protocols based on

[5]Full documentation and source codes are available at http://people.irisa.fr/Alexandre.Debant/proving-physical-proximity-using-symbolic-methods.html

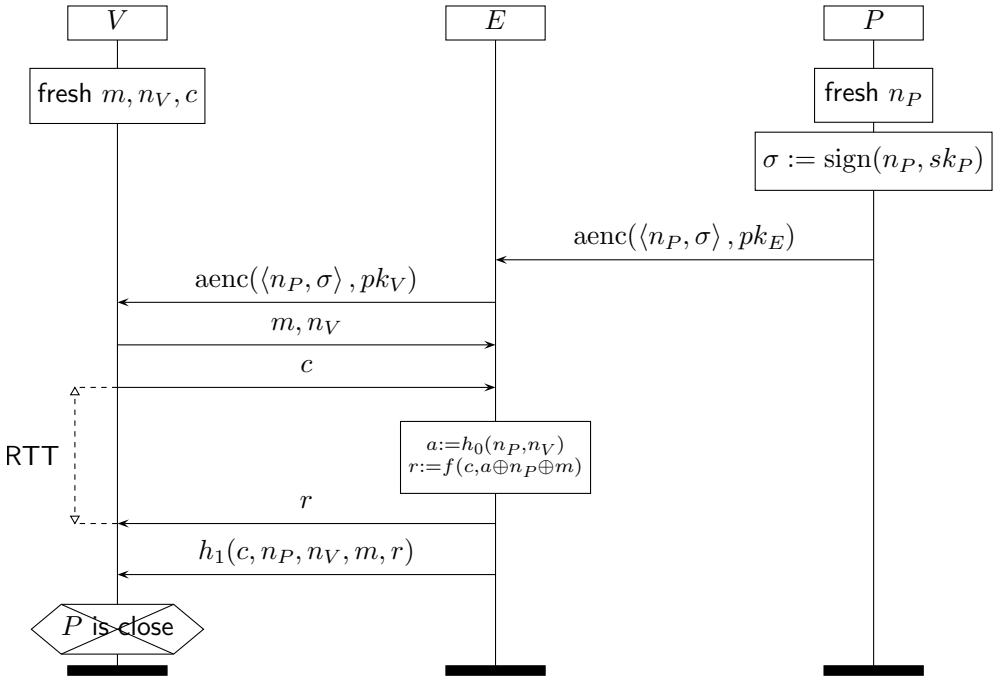


Figure 2.6 A mafia fraud attack on the SPADE protocol [BGG+16], in which the legitimate prover P is far from the verifier V, and the dishonest prover E is close to V. The prover P starts the protocol with (supposedly a verifier) E by sending $\langle n_P, \text{sign}(n_P, sk_P) \rangle$ encrypted with the public key of E, which is all E needs to impersonate P to V for the rest of the execution, thus making V believe that it is P who is close.

the ISO/IEC 14443, indeed fail to defend against both distance hijacking and distance fraud. We will also suggest simple fixes to the protocol that prevent such attacks.

Part I

Computational Analysis of Distance-Bounding Protocols

3

Lookup-Based Protocols

Distance-bounding protocols measure the round-trip times of a series of challenge/response rounds, during which the proving party must have minimal computational overhead. This can be achieved by pre-computing the responses and storing them in a constant-time access lookup table.

In this chapter we study such class of lookup-based distance-bounding protocols. By designing an aumotata-based model for these protocols, we study their properties such as security bounds in relation to space complexity. Further, we identify a family of lookup-based protocols which balance well security and space complexity.

Organization– In Section 3.1 we define an abstract model for the representation of lookup-based protocols. Security bounds and space complexity of this class are the focus of Section 3.2. In Section 3.3 we study a prominent class of lookup-based protocols, called layered and random-labeled protocols, and define an optimal mafia fraud strategy against these protocols. Later on, in Section 3.4 we describe a family of protocols whose resistance to mafia fraud converges to the optimal value, as a security parameter grows. Conclusive remarks of the chapter are given in Section 3.5.

3.1 A Model Based on Deterministic Finite Automata

We focus on a prominent class of distance-bounding protocols. This class is composed of protocols that satisfy the following two properties:

(1) During the fast phase, the responses to the challenges are looked-up from a table built up in the slow phase.

(2) The prover does not send any messages after the fast phase has been completed.

Protocols that satisfy these two properties are called *lookup-based* distance-bounding protocols, or lookup-based protocols for short. These protocols have become attractive for ubiquitous systems as a lookup operation can be done in constant time, thus the round-trip times accurately approximate twice the prover-to-verifier distance divided by the network propagation speed, regardless of the computational power of the prover. A large number of such protocols have been proposed to date, e.g. [MP08, AT09, TMA10, GAA10, KA11, MTT16a, KKBD11, TP07]. In order to study lookup-based protocols in a generic manner, we model them via a particular class of Deterministic Finite Automata (DFA).

Definition 3.1 (State-Labeled DFA). A *State-Labeled Deterministic Finite Automata* is a tuple of the form $(\Sigma, \Gamma, Q, q_0, \delta, \ell)$ where:

- Σ is a finite set of input symbols,
- Γ is a finite set of output symbols,
- Q is a finite set of states,
- $q_0 \in Q$ is the initial state,
- $\delta\colon Q \times \Sigma \to Q$ is a state-transition function,
- $\ell\colon Q \to \Gamma$ is a labeling function on the states.

Definition 3.1 above differs from traditional DFAs in two main aspects. First, it does not define final states. This is because we use it for modeling the execution of protocols that might halt at any state. Second, it includes a labeling function on the states whose output ranges over the set of output symbols Γ. Transition labels express the challenges exchanged in the protocol, whereas the state labels define the corresponding responses, which regard the lookup operations. We will make this precise later on in Definition 3.5.

Similar to traditional DFAs, we assume that the state transition function is total. However, for the sake of simplicity, when defining or drawing DFAs we will only specify the relevant transitions and, again similar to regular DFAs, we assume that specifications are completed with an implicit *trap state* that serves as the target state for all transitions that are not shown.

In the remaining of Part I of this thesis we will use, unless otherwise indicated:

- the terms *string* and *sequence* interchangeably,
- either $s_1 \cdots s_k$, or $\langle s_1, \ldots, s_k \rangle$, or $s_1 | \cdots | s_k$, or $[s_1, \ldots, s_k]$ to refer to a sequence s of k elements (the choice depends on presentation),

- s_i to refer to the i-th symbol of a sequence s,
- ε to denote the empty sequence,
- Σ and Γ to denote the universes of challenge and response symbols, respectively, and both Σ and Γ are assumed to be composed of consecutive non-negative integer numbers starting from 0 (zero),
- n to denote the number of fast phase rounds of RTT measurements, and
- $\mathbb{A}_{\Sigma,\Gamma}$ to denote the universe of all automata with Σ and Γ as their sets of input and output symbols, respectively.

Both Σ and Γ are equal to $\{0, 1\}$ for most protocols, which means that the protocol's fast phase is composed of single-bit messages. However, whenever possible, we will develop our definitions and analyses in a generic way, i.e. in terms of $|\Sigma|$, and $|\Gamma|$, and (possibly) other protocol-specific parameters.

We define the *generalized* transition and labeling functions, for a given automaton, that extend the automaton's transition and label functions; respectively. Given a sequence of symbols as input, the generalized transition function yields the state resulting from a sequence of consecutive transitions. The generalized labeling function gives the label of the state given by generalized transition function.

Definition 3.2 (Generalized Transition). Given an automaton $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$, its *generalized transition function* $state_A \colon \bigcup_{i=0}^{n} \Sigma^i \to Q$ is defined by:

$$state_A(c) = \begin{cases} q_0 & \text{if } c = \varepsilon \\ \delta(state_A(c_1 \dots c_{n-1}), c_n) & \text{if } c = c_1 \dots c_n \ . \end{cases}$$

Definition 3.3 (Generalized Labeling). Given an automaton $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$, its *generalized labeling function* $label_A \colon \bigcup_{i=1}^{n} \Sigma^i \to \Gamma$ is defined by:

$$label_A(x) = \ell(state_A(x)).$$

We model a lookup-based protocol as a set of State-Labeled DFA, where each of them describes one of the possible executions of the protocol's fast phase. The structure and labeling of the *selected* automaton follows from the calculations in the slow phase, in which, e.g. the nonces are chosen and exchanged. Consequently, every possible outcome of the slow phase results in an automaton, so the number of automata in the protocol equals the number of different outcomes of the slow phase. The execution of a protocol therefore consists of the (random) selection of one of the automata (the slow phase) and a run of this automaton consisting of an alternation of input and output symbols (the fast phase).

Definition 3.4 (Lookup-Based Protocol). A *lookup-based protocol* is a finite set $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ such that for every $A \in Proto$, the functions $state_A$ and $label_A$ are total.

The condition guarantees that no input sequence runs into an undefined point. Observe that by definition, the domain of $state_A$ equals the domain of $label_A$ in addition to the empty sequence ε. As a running example, let us consider Hancke and Kuhn's (HK) protocol [HK05] (recall its representation from Figure 1.5). Note that this in this protocol, the lookup table is a $2n$-bit sequence.

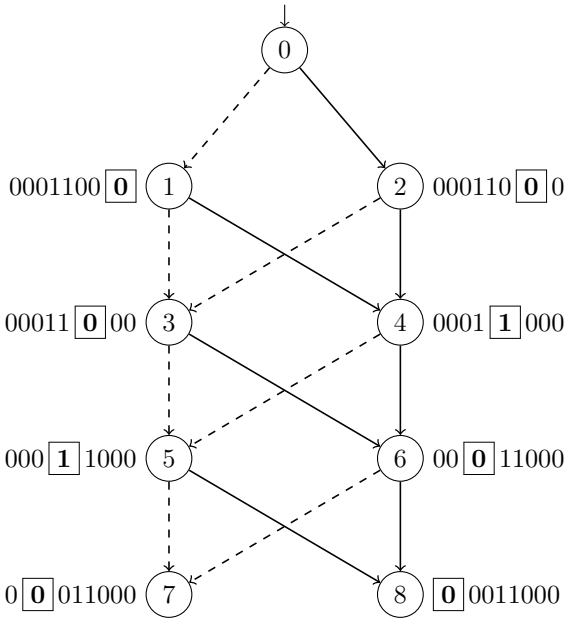


Figure 3.1 The automaton A_{24} from the HK protocol for $n = 4$ rounds.

Example 3.1 (The HK Protocol). $\mathrm{HK} = \{A_0, \ldots, A_{2^{2n}-1}\} \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ where, for every $i \in \{0, \ldots, 2^{2n} - 1\}$, $A_i = (\{0,1\}, \{0,1\}, Q, q_0, \delta, \ell_i)$ such that:

- $Q = \{0, 1, \ldots, 2n\}$,
- $q_0 = 0$,
- $\delta(q, c) = \begin{cases} q + c + 1 & \text{if } q \text{ is even} \\ q + c + 2 & \text{otherwise} , \end{cases}$
- $\ell_i(q)$ is the q-th least significant bit of i in the binary representation of i.

The number 2^{2n} of automata defining the HK protocol corresponds to the total number of possible $2n$-bit sequences that can be pre-computed. To encode the labeling functions, each bit-sequence $b_1 \ldots b_{2n} \in \{0,1\}^{2n}$ is mapped into the automaton A_i where i equals to $b_1 \ldots b_{2n}$ in decimal.

Figure 3.1 shows a graphical representation of the automaton $A_{24} \in \mathrm{HK}$, which corresponds to an execution of $n = 4$ rounds of the HK protocol with $b_1 \ldots b_{2n} = 00011000$. In this example, the states 4 and 5 are labeled with $b_4 = b_5 = 1$. The rest of the states are labeled with 0. The binary sequence 00011000 besides the states stand for 24 in binary. Dashed and solid arrows denote transitions labeled with 0 and 1, respectively.

Definition 3.5 (Execution Model). Let $\mathit{Proto} \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a lookup-based protocol. The triple $(A, c_1 \ldots c_n, r_1 \ldots r_n) \in \mathit{Proto} \times \Sigma^n \times \Gamma^n$ is a *correct execution of Proto* if $r_i = \mathit{label}_A(c_1 \ldots c_i)$ for all $i \in \{1, \ldots, n\}$.

Before the start of the fast phase, the prover and the verifier agree on a fresh automaton A. Then during the fast phase, the verifier sends n challenges $c_1 \ldots c_n$ and expects to

receive as replies the sequence $label_A(c_1) \cdots label_A(c_1 \dots c_n)$. As an example, consider again the automaton $A_{24} \in \mathrm{HK}$ depicted in Figure 3.1. Given the input bit sequence 1100, this automaton transits over the states $2, 4, 5$, and 7, whose labels are 0, 1, 1, and 0, respectively. Hence, $(A_{24}, 1100, 0110)$ is a correct execution of the HK protocol with 4 rounds. The set of all correct executions of a protocol *Proto* is denoted $[\![Proto]\!]$.

3.2 Preliminary Analysis of Lookup-Based Protocols

In this section we investigate preliminary properties of lookup-based protocols. First, we make our adversarial model explicit and define *pre-ask attacks*, which is a fundamental class of mafia fraud attacks. Hence, we prove that there does not exist a lookup-based protocol for which a mafia fraud succeeds with probability lower than $\frac{1}{2^n}(1 + \frac{n}{2})$. This result implies that the Tree protocol [AT09] is *optimal* in terms of resistance to mafia fraud. We also prove a necessary property on the probability distribution of labels for a lookup-based protocol to be optimal.

3.2.1 Formalizing Pre-Ask Attacks

A pre-ask attack is a mafia fraud attack based on the pre-ask strategy [ABK+11]. As stated in [ABK+11], this is the most effective adversary strategy to perform a mafia fraud attack against a distance-bounding protocol in which the prover does not act in the final phase. Because of this, we focus on this type of attack.

A pre-ask attack consists of two (possibly) interleaved sessions[1]: one session between the prover and the adversary –the *pre-ask session*, and another session between the adversary and the verifier. As a form of mafia fraud, the verifier and the prover are distant and the adversary's goal is to make the verifier believe the prover is close. The adversary proceeds first by relaying the verbatim messages exchanged during the slow phase, from one session to the other. Then, the adversary runs a fake fast phase with the prover by querying the latter with the adversary's own sequence of challenges –the *pre-ask challenges*. The adversary then bypasses the fast phase with the verifier, on behalf of the distant legitimate prover. To achieve this, the adversary possibly uses the messages exchanged in the pre-ask session.

Below we give a formal stochastic definition of mafia fraud, with focus on the described pre-ask strategy. We denote by $\mathbb{F}_{\Sigma,\Gamma}$ the universe of all sequences $\langle f_1, \dots, f_n \rangle$ of total functions $f_i \colon \Sigma^i \times \Sigma^n \times \Gamma^n \times \Gamma^n \to \Gamma$ for all $i \in \{1, \dots, n\}$.

Definition 3.6 (Success Probability). Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a lookup-based protocol, and for every $F = \langle f_1, \dots, f_n \rangle \in \mathbb{F}_{\Sigma,\Gamma}$ and every $x \in \Sigma^n$, let E_F^x be the event that $(A, c_1 \dots c_n, r_1 \dots r_n)$ is a correct execution of *Proto* where:

- A is a random automaton uniformly chosen from *Proto*,
- c is a random sequence uniformly chosen from Σ^n,
- $y \in \Gamma^n$ such that (A, x, y) is a correct execution of *Proto*,

[1] The usage of the term *session* here is merely illustrative, thus it does not carry the rigorousness of session identification or authentication.

- z is a random sequence uniformly chosen from Γ^n,
- $r_i = f_i(c_1 \dots c_i, x, y, z)$ for all $i \in \{1, \dots, n\}$.

Then the *probability of success of a pre-ask attack against Proto* is computed by:

$$mafia(Proto) = \max_{F \in \mathbb{F}_{\Sigma,\Gamma},\, x \in \Sigma^n} \left\{ \Pr\left(E_F^x\right) \right\}. \tag{3.1}$$

In Definition 3.6, the adversary knowledge is a correct execution (A, x, y) of *Proto* where $x_1 \dots x_n$ are the pre-ask challenges chosen by the adversary and A is randomly chosen from the set *Proto*. That is, the adversary is able to query the prover with $x_1 \dots x_n$ and receive the corresponding responses $y_1 \dots y_n$. With this knowledge, the adversary defines a strategy to respond to the verifier's challenge. We represent such strategy as a sequence $F = \langle f_1, \dots, f_n \rangle$ so the adversary's response to the verifier's i-th challenge c_i is $f_i(c_1 \dots c_i, x, y, z)$. The random sequence $z \in \Gamma^n$ is utilized for rounds in which the adversary replies randomly.

In the remaining of this Part I we will assume that the probability of success of a mafia fraud is equal to that of a pre-ask attack, in consistency with [ABK+11]. Thus, we will use indistinctly both terms to refer to the same attack.

3.2.2 Preliminary Analysis of Optimal Resistance to Mafia Fraud

The success probability of a mafia fraud attack against the Tree protocol is $mafia(\text{Tree}) = \frac{1}{2^n}\left(1 + \frac{n}{2}\right)$ [AT09]. We will prove that such probability value is indeed optimal within the class of *binary* lookup-based protocols. A lookup-based protocol *Proto* is binary if $Proto \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$, which is indeed the case of most distance-bounding protocols proposed to date.

Theorem 3.1. The probability of success of a mafia fraud attack against a binary lookup-based protocol is tightly lower-bounded by $\frac{1}{2^n}\left(1 + \frac{n}{2}\right)$.

Proof. Let $Proto \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ be a protocol and consider the following adversary strategy to execute a pre-ask attack as in Definition 3.6. Let $F = \langle f_1, \dots, f_n \rangle \in \mathbb{F}_{\{0,1\},\{0,1\}}$ where each f_i is defined by:

$$f_i(c_1 \dots c_i, x, y, z) = \begin{cases} y_i & \text{if } c_1 \dots c_i = x_1 \dots x_i \\ z_i & \text{otherwise .} \end{cases}$$

According to this strategy, at the i-th round, the adversary replies randomly unless they have guessed all verifier's challenges so far. We will prove that $mafia(Proto) \geq \frac{1}{2^n}\left(1 + \frac{n}{2}\right)$.

In effect, let $x \in \{0,1\}^n$ be the pre-ask challenge sequence, i.e. the challenges chosen by the adversary to query the prover in the pre-ask session. We will proceed to compute the probability $\Pr(E_F^x)$, recall the event E_F^x from Definition 3.6. Let M_i be the event that $c_1 \dots c_i = x_1 \dots x_{i-1}$ and that $c_i \neq x_i$ for every $i \in \{1, \dots, n\}$, where $c \in \{0,1\}^n$ is the random sequence as per the event E_F^x. From the law of total probability we have:

$$\Pr(E_F^x) = \sum_{i=1}^{n} \Pr(E_F^x \mid M_i) \Pr(M_i) + \Pr(E_F^x \mid c = x) \Pr(c = x). \tag{3.2}$$

But, given that $\Pr(M_i) = \frac{1}{2^i}$, and $\Pr(E_F^x \mid c = x) = 1$, and $\Pr(c = x) = \frac{1}{2^n}$, Equation 3.2 becomes:

$$\Pr(E_F^x) = \sum_{i=1}^{n} \Pr(E_F^x \mid M_i) \cdot \frac{1}{2^i} + \frac{1}{2^n}. \tag{3.3}$$

So far we have not used the particularity of F. Let's do so now. Indeed, if M_i occurs, then starting from the i-th round the adversary always replies randomly. This gives us $\Pr(E_F^x \mid M_i) = 1/2^{n-i+1}$ and therefore by applying this result in Equation 3.3 we obtain $\Pr(E_F^x) = \sum_{i=1}^{n} \frac{1}{2^{n-i+1}} \cdot \frac{1}{2^i} + \frac{1}{2^n} = \frac{1}{2^n}\left(1 + \frac{n}{2}\right)$. This lower bound is tight because it is realized by the Tree protocol [AT09]. □

Definition 3.7 (Optimal Binary Lookup-Based Protocol)**.** A binary lookup-based protocol is *optimal* if the probability of mafia fraud attack to succeed against it is $\frac{1}{2^n}\left(1 + \frac{n}{2}\right)$.

A necessary requirement for a binary lookup-based protocol to be optimal is that, given an input sequence x and a lookup-based protocol *Proto*, the labels assigned to the states reachable by x uniformly distribute in *Proto*.

Lemma 3.1. Let $Proto \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ be an optimal protocol, $j \in \{1, \ldots, n\}$, and $\bar{c} \in \{0,1\}^j$. Also, let S_0 and S_1 be the events that $label_A(\bar{c}) = 0$ and $label_A(\bar{c}) = 1$, respectively, for a random automaton $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$. Then, $\Pr(S_0) = \Pr(S_1) = \frac{1}{2}$.

Proof. We proceed by reduction to the absurd, i.e. let $a = \Pr(S_0)$ and assume $a \neq \frac{1}{2}$. Our goal is to reach a contradiction. We define the following pre-ask strategy $F = \langle f_1, \ldots, f_n \rangle \in \mathbb{F}_{\{0,1\},\{0,1\}}$ to execute a pre-ask attack as per Definition 3.6:

$$f_i(c_1 \ldots c_i, x, y, z_i) = \begin{cases} y_i & \text{if } c_1 \ldots c_i = x_1 \ldots x_i \\ 0 & \text{if } i = j \wedge\ c_1 \ldots c_i = \bar{c}_1 \ldots \bar{c}_i \neq x_1 \ldots x_i \wedge\ a > 1/2 \\ 1 & \text{if } i = j \wedge\ c_1 \ldots c_i = \bar{c}_1 \ldots \bar{c}_i \neq x_1 \ldots x_i \wedge\ a < 1/2 \\ z_i & \text{otherwise .} \end{cases}$$

Let $x \in \{0,1\}^n$ be the pre-ask challenges, and as in Theorem 3.1's proof, for the random $c \in \{0,1\}^n$ in the context of the event E_F^x, let M_i to be the event that $c_1 \ldots c_{i-1} = x_1 \ldots x_{i-1}$ and $c_i \neq x_i$ for every $i \in \{1, \ldots, n\}$. We proceed as in Theorem 3.1's proof until Equation 3.3:

$$\Pr(E_F^x) = \sum_{i=1}^{n} \Pr(E_F^x \mid M_i) \cdot \frac{1}{2^i} + \frac{1}{2^n}. \tag{3.4}$$

From the strategy F it follows that, for every $i \in \{1, \ldots, n\}$, $\Pr(E_F^x \mid M_i) = 1/2^{n-i+1}$ unless $c_1 \ldots c_i = \bar{c}_1 \ldots \bar{c}_i \neq x_1 \ldots x_i$. In this case, the probability of success of the adversary at the j-th round is $\max(a, 1-a) > \frac{1}{2}$. This means that $\Pr(E_F^x \mid M_i) > 1/2^{n-i+1}$ if $c_1 \ldots c_j = \bar{c}_1 \ldots \bar{c}_j \neq x_1 \ldots x_j$ or otherwise $\Pr(E_F^x \mid M_i) = 1/2^{n-i+1}$. This implies that $\Pr(E_F^x \mid M_i) > 1/2^{n-i+1}$ and by applying this result to Equation 3.4 we have:

$$\Pr(E_F^x) > \sum_{i=1}^{n} \frac{1}{2^{n-i+1}} \cdot \frac{1}{2^i} + \frac{1}{2^n} = \frac{1}{2^n}\left(1 + \frac{n}{2}\right). \tag{3.5}$$

Therefore, Equation 3.5 contradicts the assumption that *Proto* is optimal. Analogously, we prove that $\Pr(S_1) = \frac{1}{2}$. □

Lemma 3.1 along with observed structural properties of the automata composing most lookup-based protocols motivate our focus on the class of layered and random-labeled protocols. We describe and study such class of protocols in the next section.

3.3 Layered and Random-Labeled Protocols

A number of existing lookup-based protocols satisfy that, in every auomata, two input sequences reach the same state only if the sequences have the same length. Examples of such protocols are [TP07, KAK+08, HK05, AT09, MTT16a, KKBD11]. Formulated differently, two sequences of different lengths cannot reach the same state. We call the automata that satisfy this property as *layered automata*, and protocols composed of layered automata as *layered protocols*. The term layer comes from the fact that the set of states can be partitioned into n subsets (or layers) $Q_1, \ldots, Q_n$, such that the states within a layer Q_i are only reachable by input sequences of length i. We write $|x|$ to indicate the length of a sequence x.

Definition 3.8 (Layered Automaton)**.** An automaton $A \in \mathbb{A}_{\Sigma,\Gamma}$ is *layered* if:

$$\forall x, y \in \Sigma^*.\ state_A(x) = state_A(y) \implies |x| = |y|.$$

Definition 3.9 (Layered Protocol)**.** A protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ is *layered* if all its automaton are layered.

A second interesting property of lookup-based protocols we focus on is motivated by Lemma 3.1. We observe that the labeling function of the DFAs in a protocol plays an important role in the protocol's resistance to pre-ask attacks. We thus define the notion of *random-labeling*, whose intent is to maximize the adversary's uncertainty on the label of a given (even if known) state.

Definition 3.10 (Random-Labeled Protocol)**.** A protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ is *random-labeled* if for every $(\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$ and for every labeling function $\ell' \colon Q \to \Gamma$, it holds that $(\Sigma, \Gamma, Q, q_0, \delta, \ell') \in Proto$.

A random-labeled protocol accounts for all possible labeling functions that can be defined on a set of states. This property holds in most distance-bounding protocols. We show next that any binary random-labeled protocol satisfies the statement of Lemma 3.1.

Lemma 3.2. Let $Proto \in \mathbb{A}_{\{0,1\},\{0,1\}}$ be a random-labeled protocol and let $j \in \{1, \ldots, n\}$ and $\bar{c} \in \{0,1\}^j$. Define the events S_0 and S_1 as in Lemma 3.1. Then $\Pr(S_0) = \Pr(S_1) = \frac{1}{2}$.

Proof. Consider the relation $R \subseteq Proto \times Proto$ defined by $(A, A') \in R$ if and only if $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ and $A' = (\Sigma, \Gamma, Q, q_0, \delta, \ell')$ for some Q, q_0, δ, ℓ, and ℓ' in their respective domains, and:

$$\forall q \in Q \setminus \{state_A(\bar{c})\}.\ \ell(q) = \ell'(q).$$

Let B_0 (resp. B_1) be the largest subset of $Proto$ such that $label_A(\bar{c}) = 0$ (resp. $label_A(\bar{c}) = 1$) for all $A \in B_0$ (resp. $A \in B_1$). But, for all $(A, A') \in R$, it holds that

$A \in B_0 \iff A' \in B_1$, thus $|B_0| = |B_1|$ and $\{B_0, B_1\}$ is a partition of *Proto*. Hence:

$$\Pr(S_0) = \frac{|B_0|}{|Proto|} = \frac{|B_1|}{|Proto|} = \Pr(S_1) = \frac{1}{2}.$$

□

The vast majority of lookup-based protocols published to date are layered and random-labeled. Only the Poulidor [TMA10] protocol is not, to the best of our knowledge. Figure 3.1 clearly shows that the example automaton of the HK protocol is layered, because the states of the i-th layer, i.e. $2i-1$ and $2i$, with $0 < i \leq 4$, can only be reached by an input sequence of length i. The labeling function, i.e., the association from states to bits, is composed in a random way. In the remaining of this chapter we will thus focus on the analysis of layered and random-labeled lookup-based protocols.

We provide next an optimal adversary strategy to execute a pre-ask attack against a layered and random-labeled lookup-based protocol. It turns out that said strategy is simply to reply to the verifier's challenges exactly with the same responses as those obtained from the prover in the pre-ask session. Recall that the pre-ask session is the session between the prover and the adversary.

Theorem 3.2. Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a layered and random-labeled lookup-based protocol and let $x \in \Sigma^n$ be the pre-ask challenges. For every $S \in \mathbb{F}_{\Sigma,\Gamma}$, let E_S^x (as in Definition 3.6) that the pre-ask strategy S succeeds in attacking *Proto* with the pre-ask challenges x. Consider the strategy $F = \langle f_1, \ldots, f_n \rangle \in \mathbb{F}_{\Sigma,\Gamma}$ where $f_i(c, x, y, z) = y_i$ for all $i \in \{1, \ldots, n\}$. Then:

$$\Pr(E_F^x) = \max_{S \in \mathbb{F}_{\Sigma,\Gamma}} \{\Pr(E_S^x)\}.$$

Proof. Let's first introduce four functions we will use throughout this proof.

- $\delta \colon \mathbb{F}_{\Sigma,\Gamma} \to \mathcal{P}(\mathbb{N} \times \Sigma^+)$ defined by:

$$\delta(\langle s_1, \ldots, s_n \rangle) = \big\{(t, \alpha) \mid t \in \{1, \ldots, n\}, \alpha \in \Sigma^t, y, z \in \Gamma^n, \\ (A, x, y) \in [\![Proto]\!],\ y_t \neq s_t(\alpha, x, y, z)\big\}. \tag{3.6}$$

  Intuitively, $\delta(S)$ regards the sequences of challenges (not necessarily of length n) for which the strategies F and S choose different response.

- $\varepsilon \colon \mathbb{F}_{\Sigma,\Gamma} \to \mathbb{F}_{\Sigma,\Gamma}$ defined by $\varepsilon(\langle s_1, \ldots, s_n \rangle) = \langle s_1, \ldots s_{t-1}, s'_t, s_{t+1}, \ldots, s_n \rangle$ where (t, α) is the syntactically least pair from $\delta(\langle s_1, \ldots, s_n \rangle)$, and s'_t is defined by:

$$s'_t(c, x, y, z) = \begin{cases} y_t & \text{if } c = \alpha \\ s_t(c, x, y, z) & \text{otherwise}. \end{cases} \tag{3.7}$$

  Intuitively, $\varepsilon(S)$ slightly approximates the strategy S towards the strategy F.

- $\lambda \colon Proto \times \Sigma^n \times \mathbb{F}_{\Sigma,\Gamma} \times \Gamma^n \to \{0, 1\}$ defined by:

$$\lambda(A, c, \langle s_1, \ldots, s_n \rangle, z) = \begin{cases} 1 & \text{if } \forall j \leq n.\ label_A(c_1 \ldots c_j) = s_j(c_1 \ldots c_j, x, y, z) \\ & \quad \text{where } y \in \Gamma^n \text{ s.t. } (A, x, y) \in [\![Proto]\!] \\ 0 & \text{otherwise}. \end{cases}$$

- $win\colon \mathbb{F}_{\Sigma,\Gamma} \to \mathbb{N}$ defined by:

$$win(S) = \sum_{A \in Proto} \sum_{z \in \Gamma^n} \sum_{c \in \Sigma^n} \lambda(A, c, S, z).$$

  One can read the above sum as the total number of automata in *Proto* the strategy S "wins", i.e. it replies correctly all verifier's challenges. Observe that if $\delta(S) = \emptyset$ then $win(F) = win(S)$.

It is easy to observe that, for every $S \in \mathbb{F}_{\Sigma,\Gamma}$:

$$\Pr\left(E_S^x\right) = \frac{win(S)}{|\Sigma|^n |\Gamma|^n |Proto|}. \tag{3.8}$$

So, we will prove that $win(F) = \max_{S \in \mathbb{F}_{\Sigma,\Gamma}} \{win(S)\}$. We will proceed by reduction to the absurd, i.e. assume that a strategy $S \in \mathbb{F}_{\Sigma,\Gamma}$ exists such that:

$$win(F) < win(S). \tag{3.9}$$

We will reach a contradiction. In effect, let (t, α) be the syntactically least pair from $\delta(S)$. Notice that such pair exists because $\delta(S) \neq \emptyset$ or otherwise $win(F) = win(S)$ which contradicts the inequality in Equation 3.9. Further, let $R \subseteq Proto \times Proto$ be a relation defined by $(A, A') \in R$ if and only if $(A, A') \in R$ if and only if $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ and $A' = (\Sigma, \Gamma, Q, q_0, \delta, \ell')$ for some Q, q_0, δ, ℓ and ℓ' in their respective domains such that:

$$\forall q \in Q \setminus \{state_A(x_1 \dots x_t)\}.\ \ell(q) = \ell'(q).$$

Note that R is reflexive, symmetric and transitive. So, let $k \in \mathbb{N}$ and $A_1, \dots, A_k \in Proto$ be k automata such that $[A_1], \dots, [A_k]$ are the equivalence classes in *Proto* with respect to R. In addition, let $J \subseteq \{1, \dots, k\}$ be the set containing all $j \in \{1, \dots, k\}$ such that $state_{A_j}(\alpha) = state_{A_j}(x_1 \dots x_t)$.

Consider now $S' = \varepsilon(S)$, which exists because (t, α) exists. From *Proto* being layered and random-labeled, and from the definition of S', it follows that, for every $j \in \{1, \dots, k\}$, every $\beta \in \Sigma^{n-t}$, and every $z \in \Gamma^n$:

- If $j \in J$ then:

  (1) $\lambda(A_j, \alpha \cdot \beta, S', z) = \lambda(A, \alpha \cdot \beta, S', z)$ for all $A \in [A_j]$, and

  (2) If $\lambda(A, \alpha \cdot \beta, S, z) = 1$ for some $A \in [A_j]$ then $\lambda(A_j, \alpha \cdot \beta, S', z) = 1$.

- If $j \notin J$ then:

  (1) $\sum_{A \in [A_j]} \lambda(A, \alpha \cdot \beta, S, z) = \sum_{A \in [A_j]} \lambda(A, \alpha \cdot \beta, S', z) \leq 1$.

From this case analysis and the definition of S' we obtain:

$$\begin{aligned} win(S') - win(S) &= \sum_{z \in \Gamma^n} \sum_{\beta \in \Sigma^{n-t}} \sum_{j=1}^{k} \sum_{A \in [A_j]} \lambda(A, \alpha \cdot \beta, S', z) - \lambda(A, \alpha \cdot \beta, S, z) \\ &= \sum_{z \in \Gamma^n} \sum_{\beta \in \Sigma^{n-t}} \sum_{j \in J} \left(|[A_j]| \lambda(A_j, \alpha \cdot \beta, S', z) - \sum_{A \in [A_j]} \lambda(A, \alpha \cdot \beta, S, z) \right) \\ &\geq 0. \end{aligned} \tag{3.10}$$

Hence, from the inequality in Equation 3.10 and given that $\delta(S') = \delta(S) \setminus \{(t, \alpha)\}$, it follows that, from a finite number of successive applications of ε, we obtain a strategy $S'^{\cdots\prime} = \varepsilon(\cdots\varepsilon(S)\cdots)$ such that $\delta(S'^{\cdots\prime}) = \emptyset$ and $win(S'^{\cdots\prime}) \geq \cdots \geq win(S') \geq win(S)$. These two results together give us $win(F) \geq win(S)$, which contradicts Equation 3.9. □

3.4 The Family of Uniform Protocols

In Theorem 3.1 we proved that $\frac{1}{2^n}\left(1 + \frac{n}{2}\right)$ is a tight lower bound on the resistance to mafia fraud of lookup-based protocols with n rounds. The only known protocol that achieves such lower bound is the Tree protocol [AT09]. However, the automata of the Tree protocol have an exponential number of states. The number of states of the automata of a protocol defines the space complexity of the protocol, which in turn determines the memory that the protocol requires.

In this section we present a family of lookup-based protocols which have excellent resistance to pre-ask attacks in relation to their space complexity. In particular, binary protocols within this family have resistance to pre-ask attacks which converges to the above-mentioned optimal bound as a security parameter, which we call *uniformity*, converges to its maximum.

3.4.1 Uniform Protocols

Uniform protocols are layered and random-labeled protocols that satisfy an additional property: an integer value u exists such that in any automaton, two input sequences reach the same state if and only if the last u symbols if they have no less that u symbols, or otherwise the two sequences are the same. The value u is called the uniformity.

This property relates to the adversary's chance to guess the correct automaton states which an execution passes through. If so, the adversary would have certainty on the responses associated to those guessed states and in turn to the correct responses to the corresponding verifier's challenges. We formally define next the notion of uniformity.

Definition 3.11 (u-uniform Protocol)**.** Let $u \in \{1, \ldots, n\}$. A protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ is *u-uniform* (or simply *uniform*) if it is layered and random-labeled and:

$$\forall A \in Proto, k \in \{1, \ldots, n\},\ x, y \in \{0,1\}^k.$$
$$state_A(x) = state_A(y) \iff \forall i \in \{\max(1, k-u+1), \ldots, k\}.\ x_i = y_i. \tag{3.11}$$

As briefly motivated earlier, the notion of uniformity is related to the adversary's chance to predict the correct states in the pre-ask session. Suppose the adversary chooses a challenge sequence $x_1 \ldots x_n$ to query the prover and consider $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ to be the selected automaton for the protocol execution, which is unknown to the adversary. Suppose also $y_1 \ldots y_n$ are the verifier's challenges. Let's denote $q_1 \ldots q_n$ and $q'_1 \ldots q'_n$ the sequences of states reached by challenge sequences x and y, respectively. Formally, $q_i = state_A(x_1 \ldots x_i)$ and $q'_i = state_A(y_1 \ldots y_i)$ for all $i \in \{1, \ldots, n\}$. Now, the more elements $q_1 \ldots q_n$ and $q'_1 \ldots q'_n$ have in common, the more vulnerable the protocol becomes, since the adversary has the responses for those states. In the case of a u-uniform protocol, for the adversary to reach the correct state, let's say at round i, he needs to guess all the u (or i if $i \leq u$) last

verifier's challenges in advance. So, the higher the uniformity value u, the harder it becomes for the adversary to make the correct guesses. The next two lemmas show that HK [HK08] and Tree [AT09] are both uniform protocols, with uniformity 1 and n, respectively.

Proposition 3.1. The HK protocol is 1-uniform.

Proof. In this proof we will use the definition of the HK protocol provided in Example 3.1. Trivially, HK is layered and random-labeled, thus let's proceed to prove Equation 3.11 with $u = 1$.

Let $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in \text{HK}$, $k \in \{1, \ldots, n\}$ and $x, y \in \{0,1\}^n$. Let $a, b \in Q = \{0, \ldots, 2n\}$ be two states such that $a \in \mathit{state}_A(x_1 \ldots x_{k-1})$ and $b \in \mathit{state}_A(y_1 \ldots y_{k-1})$. Also, let $a', b' \in \{0,1\}$ such that $a \equiv a' \pmod 2$ and $b \equiv b' \pmod 2$. Hence,

$$\begin{aligned}
\mathit{state}_A(x) = \mathit{state}_A(y) &\iff \delta(a, x_k) = \delta(b, y_k) \\
&\iff a + x_k + 1 + a' = b + y_k + 1 + b' \\
&\iff a + x_k + 1 + a' \equiv b + y_k + 1 + b' \pmod 2 \\
&\iff 2a + x_k + 1 \equiv 2b + y_k + 1 \pmod 2 \\
&\iff x_k \equiv y_k \pmod 2 \iff x_k = y_k.
\end{aligned}$$

□

Proposition 3.2. The Tree protocol is n-uniform.

Proof. Let $\text{Tree} = \{A_0, A_1, \ldots, A_N\} \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ where $N = 2^{2^{n+1}-2} - 1$ and, for every $i \in \{0, \ldots, N\}$, $A_i = (\{0,1\}, \{0,1\}, Q, q_0, \delta, \ell_i)$ such that:

- $Q = \{0, 1, \ldots, 2^{n+1} - 2\}$,
- $q_0 = 0$,
- $\delta(q, c) = 2q + c + 1$,
- $\ell_i(q)$ is the q-th bit of the binary representation of i.

Trivially, Tree is layered and random-labeled, thus let's proceed to prove Equation 3.11 with $u = n$. Let $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in \text{Tree}$, $k \in \{1, \ldots, n\}$ and $x, y \in \{0,1\}^n$. Let $a, b \in Q$ be two states such that $a \in \mathit{state}_A(x_1 \ldots x_{k-1})$ and $b \in \mathit{state}_A(y_1 \ldots y_{k-1})$. Then:

$$\mathit{state}_A(x) = \mathit{state}_A(y) \iff \delta(a, x_k) = \delta(b, y_k) \iff 2a + x_k + 1 = 2b + y_k + 1.$$

We will proceed to prove that $2a + x_k + 1 = 2b + y_k + 1 \iff (a = b \wedge x_k = y_k)$. The implication from right- to left-hand side is trivial, so we proceed by proving the implication from left- to right-hand side. Indeed,

$$\begin{aligned}
2a + x_k + 1 = 2b + y_k + 1 &\implies 2a + x_k + 1 \equiv 2b + y_k + 1 \pmod 2 \\
&\implies x_k \equiv y_k \pmod 2 \implies x_k = y_k.
\end{aligned}$$

Hence, if $2a + x_k + 1 = 2b + y_k + 1$ and $x_k = y_k$ then $a = b$. So, from a recursive reasoning we have:

$$\begin{aligned}
&state_A(x) = state_A(y)\\
&\quad\iff state_A(x_1 \dots x_{k-1}) = state_A(y_1 \dots y_{k-1}) \wedge x_k = y_k\\
&\quad\iff state_A(x_1 \dots x_{k-2}) = state_A(y_1 \dots y_{k-2}) \wedge x_{k-1} = y_{k-1} \wedge x_k = y_k\\
&\quad\dots\\
&\quad\iff state_A(x_1) = state_A(y_1) \wedge x_2 = y_2 \wedge \dots \wedge x_{k-1} = y_{k-1} \wedge x_k = y_k\\
&\quad\iff x_1 = y_1 \wedge x_2 = y_2 \wedge \dots \wedge x_{k-1} = y_{k-1} \wedge x_k = y_k.
\end{aligned}$$

□

3.4.2 Security Analysis of Uniform Protocols

To compute the success probability of a pre-ask attack against a uniform protocol, we will use the results from Theorem 3.2. That is to say, we will compute the adversary's success probability when executing the optimal pre-ask strategy defined in Theorem 3.2. We recall that such a strategy consists of replying to the verifier's challenges with the answers received from the prover in the pre-ask session.

Theorem 3.3. Let $u \in \{1, \dots, n\}$ and a binary u-uniform protocol $Proto \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$. Then $mafia(Proto) = R_n$, where $R_0 = 1$ and for all $i \in \{1, \dots, n\}$:

$$R_i = \frac{1}{2^i} + \sum_{j=0}^{i-1} \frac{R_{i-j-1}}{2^{j+\min(u,j+1)+1}}.$$

Proof. Let $x \in \{0,1\}^n$ be the pre-ask challenges, i.e. a sequence representing the adversary's challenges to query the prover in the pre-ask phase. Consider the following events, for a random atomaton $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$ and a random sequence $c \in \{0,1\}^n$:

- for all $i \in \{1, \dots, n\}$ let S_i be the event that $y_1 \dots y_i = r_1 \dots r_i$ where $y, r \in \Gamma^n$ such that $(A, x, y) \in [\![Proto]\!]$ and $(A, c, r) \in [\![Proto]\!]$ (recall from Definition 3.5 that $[\![Proto]\!]$ is the set of all correct executions of $Proto$),
- for all $i \in \{1, \dots, n\}$ and $j \in \{0, \dots, i-1\}$, let $M_{i,j}$ is the event that $x_{i-j+1} \dots x_i = c_{i-j+1} \dots c_i \wedge x_{i-j} \neq c_{i-j}$. Note that $M_{i,0}$ becomes $x_i \neq c_i$,
- for all $i \in \{1, \dots, n\}$, let G_i is the event that $x_1 \dots x_i = c_1 \dots c_i$.

  Note that G_i occurs if none of the events $M_{i,0}$, … $M_{i,i-1}$ do. This means that $\Pr(G_i \vee M_{i,0} \vee \dots \vee M_{i,i-1}) = 1$.

Our goal is to compute the values of $\Pr(S_i)$ and in particular $\Pr(S_n)$. In effect, by the law of total probability we have:

$$\begin{aligned}
\Pr(S_i) &= \Pr(S_i \mid G_i) \Pr(G_i) + \sum_{j=0}^{i-1} \Pr(S_i \mid M_{i,j}) \Pr(M_{i,j})\\
&= \frac{1}{2^i} + \sum_{j=0}^{i-1} \Pr(S_i \mid M_{i,j}) \Pr(M_{i,j}),
\end{aligned} \tag{3.12}$$

because $\Pr(S_i \mid E_i) = 1$ and $\Pr(E_i) = \frac{1}{2^i}$. Moreover, since the sequence c is chosen randomly and its bits are independent, it follows that, for every $i \in \{1, \ldots, n\}$ and every $j \in \{1, \ldots, i-1\}$:

$$\Pr(M_{i,j}) = \Pr(x_{i-j} \neq c_{i-j}) \times \prod_{k=i-j+1}^{i} \Pr(x_k = c_k) = \frac{1}{2} \times \frac{1}{2^j} = \frac{1}{2^{j+1}}. \tag{3.13}$$

Observe that $\Pr(M_{i,0}) = \Pr(x_i \neq c_i) = \frac{1}{2}$. Now let's compute the values $\Pr(S_i \mid M_{i,j})$ for $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, i-1\}$. To develop our reasoning, let's fix $i \in \{1, \ldots, n\}$ and $j \in \{0, \ldots, i-1\}$. If $M_{i,j}$ occurs, then the input sequences $x_1 \ldots x_i$ and $c_1 \ldots c_i$ have the same last j symbols. Let's analyze now the two cases:

Case $j < u$: In this case we have $state_A(x_1 \ldots x_k) \neq state_A(c_1 \ldots c_k)$ for all $k \in \{i-j, i\}$. Hence:

$$\Pr(y_{i-j} \ldots y_i = r_{i-j} \ldots r_i \mid M_{i,j}) = \frac{1}{2^{j+1}}. \tag{3.14}$$

Case $j \geq u$: Because of the uniformity property, we have that:

$$\begin{aligned} &state_A(x_1 \ldots x_{i-j}) \neq state_A(c_1 \ldots c_{i-j}),\\ &state_A(x_1 \ldots x_{i-j+1}) \neq state_A(c_1 \ldots c_{i-j+1}),\\ &\ldots\\ &state_A(x_1 \ldots x_{i-j+u-1}) \neq state_A(c_1 \ldots c_{i-j+u-1}), \end{aligned} \tag{3.15}$$

and

$$\begin{aligned} &state_A(x_1 \ldots x_{i-j+u}) = state_A(c_1 \ldots c_{i-j+u}),\\ &state_A(x_1 \ldots x_{i-j+u+1}) = state_A(c_1 \ldots c_{i-j+u+1}),\\ &\ldots\\ &state_A(x_1 \ldots x_i) = state_A(c_1 \ldots c_i). \end{aligned} \tag{3.16}$$

From the set of inequalities in Equation 3.15 and the set of equalities in Equation 3.16 and given that *Proto* is random-labeled we derive that, for every $k \in \{i-j, \ldots, i\}$:

$$\Pr(y_k = r_k \mid M_{i,j}) = \begin{cases} 1/2 & \text{if } k \leq i-j+u-1 \\ 1 & \text{otherwise}, \end{cases}$$

which leads to:

$$\Pr(y_{i-j} \ldots y_i = r_{i-j} \ldots r_i \mid M_{i,j}) = \frac{1}{2^u}. \tag{3.17}$$

Now, the event S_{i-j-1} and the event that $y_{i-j} \ldots y_i = r_{i-j} \ldots r_i$ are independent, due to the uniformity property and $x_{i-j} \neq c_{i-j}$. This gives us:

$$\Pr(S_i \mid M_{i,j}) = \Pr(S_{i-j-1} \mid M_{i,j}) \times \Pr(y_{i-j} \ldots y_i = r_{i-j} \ldots r_i \mid M_{i,j}). \tag{3.18}$$

Hence, Equations 3.14, 3.17, and 3.18 give us:

$$\Pr(S_i \mid M_{i,j}) = \frac{\Pr(S_{i-j-1} \mid M_{i,j})}{2^{\min(u,j+1)}}. \tag{3.19}$$

Furthrmore, the events $M_{i,j}$ and S_{i-j-1} are independent, thus $\Pr(S_{i-j-1} \mid M_{i,j}) = \Pr(S_{i-j-1})$. By applying this in Equation 3.19 we obtain:

$$\Pr(S_i \mid M_{i,j}) = \frac{\Pr(S_{i-j-1})}{2^{\min(u,j+1)}}. \tag{3.20}$$

From Equations 3.12, 3.13 and 3.20 and by applying the substitution $R_i = \Pr(S_i)$ we obtain the expected recursive formula. Finally, notice that $\Pr(S_i)$ is independent from x, so this observation together with Definition 3.6 and Theorem 3.2 give us $\mathit{mafia}(\mathit{Proto}) = \Pr(S_n) = R_n$. □

A consequence of this theorem is that, in uniform protocols, the adversary has no advantage in selecting the challenges to query the prover. In the next corollaries we show, by using the previous theorem, a security computation in terms of pre-ask attacks for the mentioned HK and Tree protocols.

Corollary 3.1. Consider the HK protocol as defined in Example 3.1. Then:

$$\mathit{mafia}(\mathrm{HK}) = \left(\frac{3}{4}\right)^n. \tag{3.21}$$

Proof. Since the HK protocol is 1-uniform (see Proposition 3.1), we have $R_i = \frac{1}{2^i} + \sum_{j=0}^{i-1} \frac{R_{i-j-1}}{2^{j+2}}$. Thus, by multiplying this equation by 2^i we obtain that:

$$2^i R_i = 1 + \sum_{j=0}^{i-1} 2^{i-j-2} R_{i-j-1} = 1 + \frac{1}{2} \sum_{j=0}^{i-1} 2^{i-j-1} R_{i-j-1}.$$

This can be written as $2^i R_i = 1 + \frac{1}{2} \sum_{k=0}^{i-1} 2^k R_k$ since $i - j - 1$ goes from 0 to $i - 1$. By applying now the substitution $B_i = 2^i R_i$ we obtain $B_i = 1 + \frac{1}{2} \sum_{k=0}^{i-1} B_k$. Hence, from this last result we derive $B_{i+1} - B_i = \frac{1}{2} B_i$ and thus $B_{i+1} = \frac{3}{2} B_i$. This implies that $B_{i+1} = \frac{3}{2} B_i$ and given that $B_0 = 1$, we have $B_i = \left(\frac{3}{2}\right)^i$. Therefore $R_i = \frac{1}{2^i} B_i = \left(\frac{3}{4}\right)^i$. □

Corollary 3.2. Consider the Tree protocol as defined in Proposition 3.2. Then:

$$\mathit{mafia}(\mathrm{Tree}) = \frac{1}{2^n} \left(1 + \frac{n}{2}\right).$$

Proof. Since the Tree protocol is n-uniform (see Proposition 3.2), we have $R_i = \frac{1}{2^i} + \sum_{j=0}^{i-1} \frac{R_{i-j-1}}{2^{2j+2}}$. By multiplying this equation by 4^i we obtain that:

$$4^i R_i = 2^i + \sum_{j=0}^{i-1} 4^{i-j-1} R_{i-j-1} = 2^i + \sum_{k=0}^{i-1} 4^k R_k.$$

By letting $B_i = 4^i R_i$ in the equation above we have $B_i = 2^i + \sum_{k=0}^{i-1} B_k$. Therefore, from this last result we derive that $B_{i+1} - B_i = 2^i + B_i$ and consequently $B_{i+1} = 2B_i + 2^i$ and $\frac{B_{i+1}}{2^{i+1}} = \frac{B_i}{2^i} + \frac{1}{2}$. Now, by letting $D_i = \frac{B_i}{2^i}$ we have that $D_{i+1} = D_i + \frac{1}{2}$ and therefore $D_i = \frac{i}{2} + D_0$. Since $R_0 = 1$, we have that $B_0 = D_0 = 1$. Therefore, $B_i = 2^i(1 + \frac{i}{2})$ which implies that $R_i = \frac{1}{2^i}(1 + \frac{i}{2})$. □

Next we prove that the resistance to pre-ask attacks of uniform protocols monotonically depends on their uniformity value.

Theorem 3.4. Let $u, v \in \{1, \ldots, n\}$ with $u \leq v$ and let $P_u, P_v \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ be a u-uniform and a v-uniform protocols, respectively. Then $\mathit{mafia}(P_u) \geq \mathit{mafia}(P_v)$.

Proof. To refer to the recursive equation in Theorem 3.3 for P_u, we introduce the notation:

$$R_i^u = \frac{1}{2^i} + \sum_{j=0}^{i-1} \frac{R_{i-j-1}^u}{2^{j+\min(u,j+1)+1}} \tag{3.22}$$

Analogously we use R_i^v for P_v. We proceed by induction over i to prove that $R_i^u \geq R_i^v$ for every $i \in \{0, \ldots, n\}$ and in particular that $R_n^u \geq R_n^v$.

- *Base Case:* $R_0^u \geq R_0^v$.

  It trivially holds, given that $R_0^u = R_0^v = 1$.

- *Induction Hypothesis:* $\forall j < i.\ R_j^u \geq R_j^v$.

- *Induction Thesis:* $R_i^u \geq R_i^v$.

  From $u \leq v$ it follows that $\min(u, j+1) \leq \min(v, j+1)$ for all $j \in \{1, \ldots, n\}$ and consequently $1/2^{j+\min(u,j+1)+1} \geq 1/2^{j+\min(v,j+1)+1}$. Therefore, from this and the the induction hypothesis we complete the proof $R_i^u \geq R_i^v$.

□

Let $\pi\colon \{1, \ldots, n\} \to \left[\frac{1}{2^n}(1 + \frac{n}{2}), \left(\frac{3}{4}\right)^n\right]$ be a function such that $\pi(u) = \mathit{mafia}(\mathit{Proto}_u)$ where $\mathit{Proto}_u \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ is a (binary) u-uniform protocol. Theorem 3.4 proves that π is decreasing and approaches $\frac{1}{2^n}\left(1 + \frac{n}{2}\right)$ when u approaches n. Based on this, we assert that the closer the uniformity value gets to n (resp. 1) the lower (resp. higher) the success probability of a pre-ask attack. In particular, n-uniform protocols (such as the Tree protocol) are optimal within this class, whereas 1-uniform (such as the HK protocol) perform worst.

3.4.3 Constructing a Uniform Protocol

In this section we provide an automata-based construction of a binary u-uniform protocol, for an arbitrary $u \in \{1, \ldots, n\}$. We prove that our construction indeed satisfies the definition of uniformity. We also describe the proposed protocol in standard cryptographic notation. Thus, this confirms that we can always build a protocol with mafia fraud resistance arbitrarily close to optimal, by setting up its uniformity value and using our model.

Let u be an arbitrary number from the set $\{1, \ldots, n\}$. The proposed u-uniform protocol is the set $\{A_0, A_2, \ldots, A_N\} \subseteq \mathbb{A}_{\{0,1\},\{0,1\}}$ where $N = 2^{(n-u+2)2^u-2} - 1$ and for every $i \in \{0, \ldots, N\}$, $A_i = (\{0,1\}, \{0,1\}, Q, q_0, \delta, \ell_i)$ such that:

- $Q = \{(i, d) \mid 0 \leq i \leq n \ \wedge\ 0 \leq d < \min(2^u, 2^i)\}$,
- $q_0 = (0, 0)$,

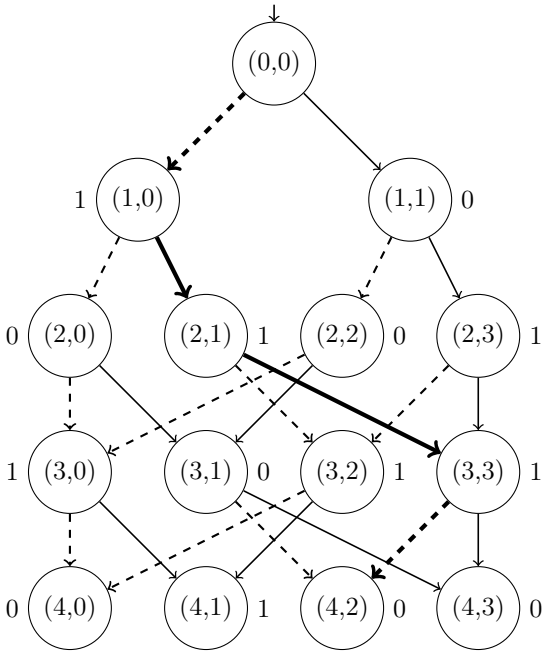


Figure 3.2 An automaton representing an instance of a 2-uniform protocol with $n = 4$ rounds. Dashed and solid arrows represent transitions when the input symbol is 0 and 1, respectively. An execution with challenge sequence 0110 whose responses are 1110 is highlighted in bold.

- for every $c \in \{0,1\}$ and $(i,d) \in Q$ such that $i < n$, $\delta((i,d),c) = (i+1,d')$ such that $d' \in \{0,\dots,2^u-1\}$ with $d' \equiv 2d + c \pmod{2^u}$,
- for every $j \in \{1,\dots,N\}$, $\ell_j(q)$ is the k-th least significant bit in the binary representation of j and k is the position of q in $Q \setminus \{q_0\}$.

In this construction, Q is a set of pairs of integers. Each pair (i,d) represents the d-th state in layer i. For two binary strings to share the last u bits, their decimal representations have to leave the same remainder when divided by 2^u. Based on this property, we build our state transition function. The expression $\delta((i,d),c) = (i+1,(2d+c) \pmod{2^u})$ stands for this idea. Notice that $dec(c_1 \dots c_i) = 2dec(c_1 \dots c_{i-1}) + c_i$, where $dec(.)$ converts binary strings into their corresponding natural number. Moreover, for every $j \in \{1,\dots,u\}$ and every $c \in \{0,1\}^j$, the values $dec(c)$ have at most 2^j different remainders when divided by 2^u, this explains our upper bound for d in the definition of Q. The last two rules in our construction represent the random-labeling property, though in practice this is achieved by generating a distribution of random bits over the set of states. An example of an automaton for a 2-uniform protocol following the above construction is depicted in Figure 3.2.

Lemma 3.3. The proposed protocol is u-uniform.

Proof. Let's denote by *Proto* the proposed protocol. We first prove that *Proto* is layered and random-labeled. Because of the definition of δ, for every pair $x, y \in \{0,1\}^i \times \{0,1\}^j$ with $i \neq j$ we have that $state_A(x) = (i, a_x)$ and $state_A(y) = (j, a_y)$, where a_x and a_y are

integer numbers (they are irrelevant here). Then $(i, a_x) \neq (j, a_y)$ and $state_A(x) \neq state_A(y)$. Therefore *Proto* is layered.

To show the random-labeling property, consider any labeling function $\ell\colon Q \setminus \{q_0\} \to \{0,1\}$. Let $B = b_1 b_2 \dots b_{|Q|-1}$ be a binary string such that $b_i = \ell(q_i)$ for every $i \in \{1, \dots, |Q|-1\}$. We recall that q_0 does not require a label since it is never used for replies. Let $D = \sum_{i=1}^{|Q|-1} 2^{i-1} b_i$ be an integer number, i.e. the decimal representation of B. Then, given that $N = 2^{|Q|-1} - 1$ we have that $D \in \{0, \dots, N\}$ and, because of our construction, $\ell_D = \ell$. This proves that for any ℓ, it holds that $(\{0,1\}, \{0,1\}, Q, q_0, \delta, \ell) \in Proto$. Every labeling function is unique since it is related to a unique integer number in D. This states that *Proto* is indeed a set.

We will proceed now to prove that *Proto* satisfies the uniformity property, i.e. Equation 3.11. Let $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$ and $k \in \{1, \dots, n\}$ and $x, y \in \{0,1\}^k$. Because of our definition of δ, we derive that $state_A(x) = (k, S_x \pmod{2^u})$ and $state_A(y) = (k, S_y \pmod{2^u})$ where:

$$S_x = \sum_{i=1}^{k} 2^{k-i} x_i \qquad \text{and} \qquad S_y = \sum_{i=1}^{k} 2^{k-i} y_i.$$

Note that S_x and S_y are the decimal representations of the bit strings x and y, respectively. Hence, $state_A(x) = state_A(y) \iff S_x \equiv S_y \pmod{2^u}$. We will analyze now two cases:

Case $k \leq u$: We have $S_x < 2^u$ and $S_y < 2^u$ and therefore:

$$state_A(x) = state_A(y) \iff S_x \equiv S_y \pmod{2^u} \iff S_x = S_y \iff x = y.$$

Case $k > u$: We can write S_x (and analogously S_y) in the following way:

$$S_x = \sum_{i=1}^{k-u} 2^{k-i} x_i + \sum_{i=k-u+1}^{k} 2^{k-i} x_i = 2^u \sum_{i=1}^{k-u} 2^{k-i-u} x_i + \sum_{i=k-u+1}^{k} 2^{k-i} x_i.$$

Since $k - i - u \geq 0$ for all $i \in \{1, \dots, k-u\}$, the elements in the first sum are integers. This implies that $S_x \equiv S'_x \pmod{2^u}$ and $S_y \equiv S'_y \pmod{2^u}$, where:

$$S'_x = \sum_{i=k-u+1}^{k} 2^{k-i} x_i \qquad \text{and} \qquad S'_y = \sum_{i=k-u+1}^{k} 2^{k-i} y_i.$$

Therefore $state_A(x) = state_A(y) \iff S_x \equiv S_y \pmod{2^u} \iff S'_x \equiv S'_y \pmod{2^u}$. Given that $k - i < u$ for every $i \in \{k-u+1, \dots, k\}$, we deduce that both S'_x and S'_y are smaller than 2^u. This implies that:

$$state_A(x) = state_A(y) \iff S'_x = S'_y \iff x_{k-u+1} \dots x_k = y_{k-u+1} \dots y_k.$$

Hence, from both cases we derive the necessary and sufficient condition stated in Definition 3.11. □

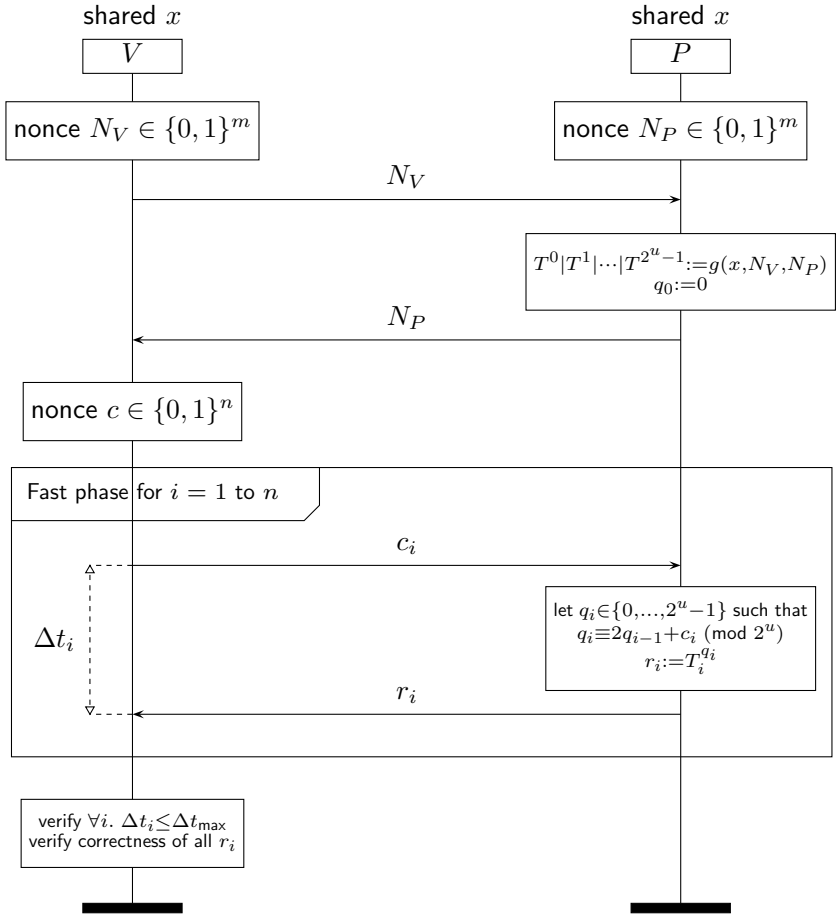


Figure 3.3 A binary u-uniform protocol. This construction takes $2^u n$ bits of memory.

We provide next a cryptographic construction for the proposed binary u-uniform distance-bounding protocol. See Figure 3.3 for a compact, high-level description of the protocol.

1. *Initialization phase:* The prover and the verifier agree on the following parameters: a shared key x, an integer number $m > 0$ which represents the length of the nonces, the number n of fast phase rounds, an integer number $u \in \{1, \ldots, n\}$ which represents the uniformity value, a pseudo-random function g, and a threshold $\Delta t_{\max}$ for the round-trip times.

2. *Slow phase:* Both parties generate nonces, $N_{\mathcal{P}}$ for the prover and $N_{\mathcal{V}}$ for the verifier. The value $N_{\mathcal{V}}$ is sent to the prover which constructs the labeling function from $g(x, N_{\mathcal{V}}, N_{\mathcal{P}})$. Then, the prover sends the nonce $N_{\mathcal{P}}$ to the verifier and the latter also computes the function $g(x, N_{\mathcal{V}}, N_{\mathcal{P}})$ to agree with the prover on the labeling function. The shared pseudo-random function g outputs 2^u (concatenated) registers $T^0|T^1|\ldots|T^{2^u-1}$ of n bits each (numbered from 1 to n, i.e. $T^j = T_1^j \cdots T_n^j$). The initial state q_0 is set to 0.

3. *Fast phase:* This phase is composed of n rounds numbered from 1 to n. At the i-th round, the verifier sends a challenge bit c_i to the prover who, upon reception,

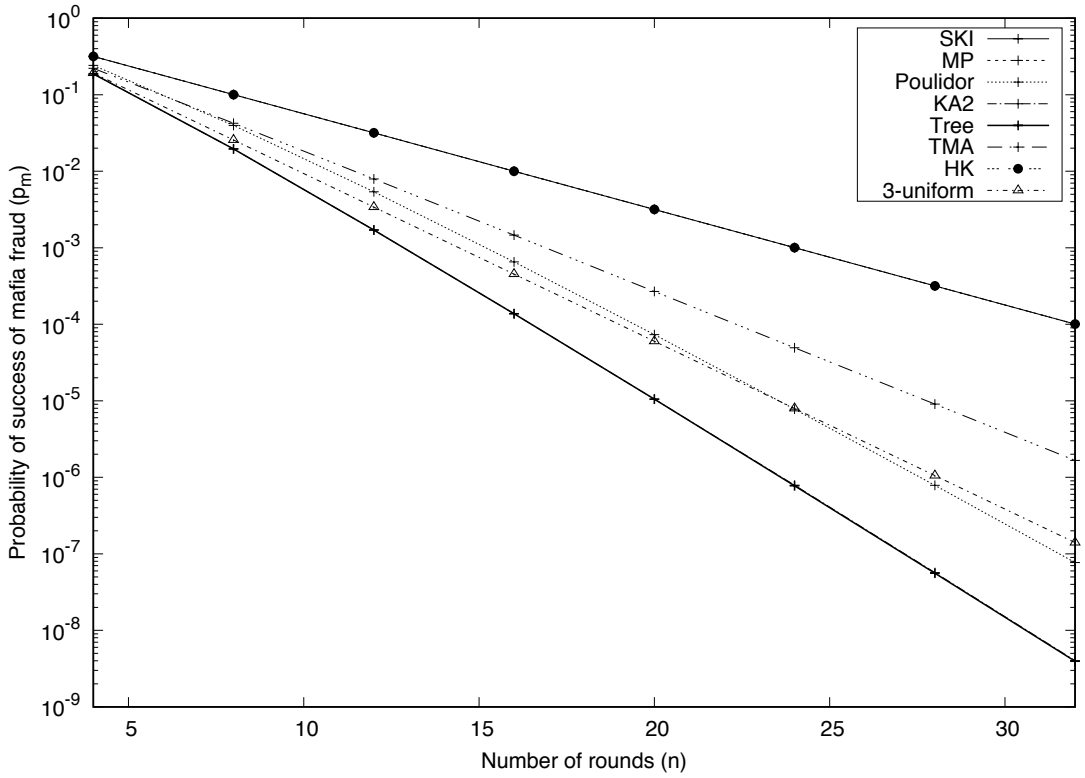


Figure 3.4 The probability of success of mafia fraud attacks against various protocols, for up to 32 rounds.

moves from the q_{i-1}-th state of layer $i-1$ to the q_i-th state of layer i such that $q_i \in \{0, \ldots, 2^u - 1\}$ with $q_i \equiv 2q_{i-1} + c_i \pmod{2^u}$ and replies with the bit $T_i^{q_i}$.

4. *Verification phase:* The protocol succeeds if and only if all the exchange times are not greater than the predefined RTT threshold $\Delta t_{\max}$ and all the responses are correct.

In order to illustrate the good balance offered by the uniform protocols in terms of security and space complexity, we depict in Figure 3.4 a chart with the probability of success of mafia fraud attacks against various distance-bounding protocols, as given by [AMT15]. The protocols are SKI [BMV13a], Tree [AT09], Poulidor [TMA10], Kim and Avoine's (KA2) [KA11], Munilla and Peinado's (MP) [MP08], Trujillo et al.'s (TMA) [TMA14], HK [HK05], and a binary 3-uniform protocol. Our protocol clearly shows competitive levels of resistance to mafia fraud attacks, without demanding exponential space like the Tree protocol. Indeed, as per the construction in Figure 3.3, the 3-uniform protocol requires $8n$ bits of memory, which is considerably smaller than $2^{n+1} - 2$ of the Tree protocol.

3.5 Conclusions

In this chapter we have introduced an abstract model for the description of lookup-based distance-bounding protocols. The model represents a protocol as a set of State-Labeled Deterministic Finite Automata and executions are represented by a random walk through

a randomly selected automata from the set. The model is sufficiently expressive to describe many published protocols, which include the well-known HK and Tree protocols.

The virtue of this model is that it supports generic analysis. For instance, we can analyze the security limits of a protocol in relation to the number of rounds of the protocol's fast phase. In particular, we analyze security in terms of mafia fraud attacks based on the pre-ask strategy. We proposed a concrete pre-ask strategy that is optimal for a prominent class of lookup-based protocols.

We introduced the notion of *uniformity*, which expresses that randomly chosen walks through the automaton have no bias towards a particular automaton state. This property indeed relates to the resistance of the protocol to mafia fraud attacks based on the pre-ask strategy. We also described a family of uniform protocols, and showed that indeed these protocols feature excellent resistance to mafia fraud even with relatively small memory usage.

4

Optimality in Lookup-Based Protocols

In Chapter 3 we showed that lookup-based protocols suffer from a trade-off between security and space complexity. For example, Avoine and Tchamkerten's protocol [AT09] *offers the optimal resistance to mafia fraud amongst all binary lookup-based protocols. However, its exponential space complexity makes it not suitable for resource-constrained systems.*

In this chapter we study this security and space complexity trade-off problem for a non-trivial class of lookup-based protocols. As a result, we construct a protocol that strikes the optimal resistance to mafia fraud within such class, for a given upper bound on the protocol size, which in turn relates to space complexity. By using a multi-criteria comparison method, we show that our protocol compares well with protocols outside the analyzed class.

Organization– In Section 4.1 we introduce two binary relations on automata that allow for description of most (if not all) existing lookup-based distance-bounding protocols. In Section 4.2 we use the two relations to formalize the security and size trade-off optimality problem for layered and random-labeled protocols. Later in Section 4.3 we propose a protocol solution for such a problem, and give a cryptographic specification of the proposed protocol protocol. Section 4.4 presents a comparison of the proposed protocol with existing protocols, which are not layered or even not lookup-based. Section 4.5 summarizes the findings of this chapter.

4.1 Equivalence Relations between Automata

By observing the underlying structure of the automata of various lookup-based protocols, one can notice structural similarities between them. For example, any pair of automata of the HK protocol [HK05] only differ in the labels assigned to the states. Such "pattern" in the automata of a protocol can be described through equivalence relations.

In this section we describe two relations on State-Labeled DFAs and show that these relations are sufficiently expressive to specify all (to the best of our knowledge) lookup-based protocols by using a *closure* operator on sets of automata, with respect to one of the two relations. Moreover, by using the automata relations, we produce closed formulas of success probability of pre-ask attacks against protocols that are specified with the aforementioned closure operator and the relations.

4.1.1 Defining the Relations

Most lookup-based protocols proposed to date satisfy that any pair of automata differ only in their labeling functions. This property, which we call *state-label-insensitive*, is defined as a binary relation on automata. Examples of protocols in which every pair of their automata are related according to the state-label-insensitive relation are HK [HK05], Kim and Avoine's [KA11], Tree [AT09], and Kardas et al.'s PUF [KKBD11]. The state-label-insensitive relation makes the design and implementation of such protocols simple in that the verifier and prover agree on the structure of the lookup table, whilst only requiring randomness on the pre-computed values.

Definition 4.1 (Relation $\mathcal{S}$)**.** The *state-label-insensitive* relation $\mathcal{S} \subseteq \mathbb{A}_{\Sigma,\Gamma} \times \mathbb{A}_{\Sigma,\Gamma}$ is defined by $(A, A') \in \mathcal{S}$ if and only if $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ and $A' = (\Sigma, \Gamma, Q, q_0, \delta, \ell')$ for some Q, q_0, δ, ℓ and ℓ' in their respective domains.

The Poulidor protocol [TMA10] is the only published lookup-based protocol that does not satisfy that every pair of automata in it are related according to $\mathcal{S}$. In this protocol, the authors designed a mechanism to prevent (to some extent) an adversary from knowing which state is being used by the prover at any round of the fast phase. The idea is that the probability of two automata sharing the same transition function must be negligible. Such mechanism seems to improve the resistance to pre-ask attacks as the adversary cannot easily use knowledge acquired in previous rounds to succeed in the current round of the fast phase.

While two automata in the Poulidor protocol can have different transition functions, they still preserve a slightly weaker structural property than the above mentioned state-label-insensitive property. If we ignore the edge labels of the automata –only look at the structure of the underlying graph of the automata– the transition functions of the automata in the Poulidor protocol are identical. We provide a formal definition for this structural relation next.

Definition 4.2 (Relation $\mathcal{L}$)**.** The *label-insensitive* relation $\mathcal{L} \subseteq \mathbb{A}_{\Sigma,\Gamma} \times \mathbb{A}_{\Sigma,\Gamma}$ is defined by $(A, A') \in \mathcal{L}$ if and only if $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ and $A' = (\Sigma, \Gamma, Q, q_0, \delta', \ell')$ for some Q, q_0, δ, δ', ℓ and ℓ' in their respective domains such that for every $q \in Q$, a bijective function $\sigma\colon \Sigma \to \Sigma$ exists such that $\delta(q, c) = \delta'(q, \sigma(c))$ for all $c \in \Sigma$.

Observe that both relations $\mathcal{S}$ and $\mathcal{L}$ are reflexive, symmetric and transitive. This property will be used when developing our security analyses. In addition, we will be using a fundamental operator on sets with respect to binary relations –the closure.

Definition 4.3 (Closure). Given a protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ and a relation $\mathcal{R} \subseteq \mathbb{A}_{\Sigma,\Gamma} \times \mathbb{A}_{\Sigma,\Gamma}$, the *closure* of $Proto$ with respect to $\mathcal{R}$, denoted by $Proto^{\mathcal{R}}$, is the smallest superset of $Proto$ such that:

$$\forall A \in Proto^{\mathcal{R}}, A' \in \mathbb{A}_{\Sigma,\Gamma}.\ (A, A') \in \mathcal{R} \implies A' \in Proto^{\mathcal{R}}.$$

We say that a protocol $Proto$ is *closed with respect to a relation* $\mathcal{R}$ if $Proto = Proto^{\mathcal{R}}$. Furthermore, if $\forall A, A' \in Proto.\ (A, A') \in \mathcal{R}$, then we say that $Proto$ is *consistent with respect to* $\mathcal{R}$. The Poulidor protocol, for example, is closed with respect to both $\mathcal{S}$ and $\mathcal{L}$, whereas it is consistent only with respect to $\mathcal{L}$.

Example 4.1 (The Poulidor Protocol). $\text{Poulidor} = \{(\{0,1\}, \{0,1\}, Q, q_0, \delta, \ell)\}^{\mathcal{L}}$ where:

- $Q = \{0, 1, \ldots, 2n-1\}$,
- $q_0 = 0$,
- $\delta(q, c) = q' \in \{0, \ldots, 2n-1\}$ such that $q' \equiv q + c + 1 \pmod{2n}$,
- ℓ is any function from Q to $\{0, 1\}$.

The vast majority of lookup-based distance-bounding protocols published to date, if not all, can be modeled by using the closure operator. A protocol being closed under $\mathcal{S}$ is equivalent to being random-labeled (Chapter 3, Section 3.3, Definition 3.10).

4.1.2 Security Analysis through the Relations

In Theorem 3.2, we provided a deterministic, optimal pre-ask strategy to execute a mafia fraud against a layered and random-labeled lookup-based distance-bounding protocol. Such strategy consists of simply replying to the verifier's challenges with the exactly the same sequence of responses obtained from the prover in the pre-ask session.

Proposition 4.1 (Probability of Success of Mafia Fraud). Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a layered protocol that is closed under $\mathcal{S}$. For every $x \in \Sigma^n$, let E^x be the event that, for a random sequence $c \in \Sigma^n$ and a random automaton $A \in Proto$, (A, x, y) and (A, c, y) are both correct executions of $Proto$, for some $y \in \Gamma^n$. Then:

$$mafia(Proto) = \max_{x \in \Sigma^n} \{\Pr(E^x)\}.$$

Proof. The proof follows straightforwardly from Definition 3.6 and Theorem 3.2. □

In Lemmas 4.1 and 4.2 below, we make Proposition 4.1 more precise by providing concrete formulas to compute the probability of success of (pre-ask strategy) mafia fraud against protocols that are consistent and closed with respect to $\mathcal{S}$ and $\mathcal{L}$, respectively. We do so by considering, for a given automaton, the meeting points between the input sequence used by the adversary during the pre-ask session and the challenges sent by the verifier.

Definition 4.4 (Meeting Points). Given an automaton $A \in \mathbb{A}_{\Sigma,\Gamma}$ and two input sequences $x, c \in \Sigma^n$, the *set of meeting points of x and c* is defined as follows:

$$meet_A(x, y) = \{i \in \{1, \ldots, n\} \mid state_A(x_1 \ldots x_i) = state_A(c_1 \ldots c_i)\}.$$

Lemma 4.1. Let $A \in \mathbb{A}_{\Sigma,\Gamma}$ be a layered automaton, then:

$$mafia\left(\{A\}^{\mathcal{S}}\right) = \frac{1}{|\Sigma|^n |\Gamma|^n} \max_{x \in \Sigma^n} \left\{ \sum_{c \in \Sigma^n} |\Gamma|^{|meet_A(x,c)|} \right\}.$$

Proof. Let $Proto = \{A\}^{\mathcal{S}}$ and for every $x, c \in \Sigma^n$ define the event E_c^x that, for a random automaton $A' \in Proto$, both (A', x, y) and (A', c, y) are correct executions of $Proto$, for some $y \in \Gamma^n$. Because $Proto$ is closed under $\mathcal{S}$, for all $x, c \in \Sigma^n$, the following two properties hold:

- $meet_A(x, c) = meet_{A'}(x, c)$ for all $A' \in Proto$;
- there are $|\Gamma|^{|Q|-(n-|meet_A(x,c)|)}$ automata $A' \in Proto$ such that both (A', x, y) and (A', c, y) are correct executions of $Proto$ for some $y \in \Gamma^n$.

Therefore, for all $x, c \in \Sigma^n$, it holds that:

$$\Pr(E_c^x) = \frac{|\Gamma|^{|Q|-(n-|meet_A(x,c)|)}}{|\Gamma|^{|Q|}} = \frac{|\Gamma|^{|meet_A(x,c)|}}{|\Gamma|^n}. \tag{4.1}$$

As in Proposition 4.1, for every $x \in \Sigma^n$ define the event E^x that E_c^x occurs, for a random $c \in \Sigma^n$. Thus:

$$\Pr(E^x) = \frac{1}{|\Sigma|^n} \sum_{c \in \Sigma^n} \Pr(E_c^x) = \frac{1}{|\Sigma|^n |\Gamma|^n} \sum_{c \in \Sigma^n} |\Gamma|^{|meet_A(x,c)|}. \tag{4.2}$$

Finally, from Equation 4.2 and Proposition 4.1 we obtain the expected result. □

Lemma 4.2. Let $A \in \mathbb{A}_{\Sigma,\Gamma}$ be a layered automaton, then:

$$mafia\left(\{A\}^{\mathcal{L}}\right) = \frac{1}{|\Sigma|^{2n} |\Gamma|^n} \sum_{x,c \in \Sigma^n} |\Gamma|^{|meet_A(x,c)|}.$$

Proof. Let $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ and let $Proto = \{A\}^{\mathcal{L}}$. As in Proposition 4.1, for every $x \in \Sigma^n$ define the the event E^x that, a random automaton $A' \in Proto$ and a random input sequence $c \in \Sigma^n$, both (A', x, y) and (A', c, y) are correct executions of $Proto$, for some $y \in \Gamma^n$.

Let $k \in \mathbb{N}$ and $A_1, \ldots, A_k \in Proto$ such that $[A_1], \ldots, [A_k]$ are the equivalence classes of $Proto$ with respect to $\mathcal{S}$. Further, let $x \in \Sigma^n$ and let A' be the random automaton in the context of the event E^x. According to the law of total probability we have:

$$\Pr(E^x) = \sum_{i=1}^{k} \Pr\left(E^x \mid A' \in [A_i]\right) \Pr(A' \in [A_i]) = \frac{1}{k} \sum_{i=1}^{k} \Pr(E^x \mid A' \in [A_i]). \tag{4.3}$$

Define now, for every $i \in \{1, \ldots, k\}$, the set $R_i = \{c \in \Sigma^n \mid \forall j \leq n.\ state_A(x_1 \ldots x_j) = state_{A_i}(c_1 \ldots c_j)\}$. Therefore:

$$\Pr(E^x \mid A' \in [A_i]) = \Pr(E^c \mid A' \in [A]) \tag{4.4}$$

for all $i \in \{1, \ldots, k\}$ and all $c \in R_i$. Hence, by iterating i over $\{1, \ldots, k\}$ and c on the corresponding R_i we obtain:

$$\sum_{i=1}^{k} \sum_{c \in R_i} \Pr(E^x \mid A' \in [A_i]) = \sum_{i=1}^{k} \sum_{c \in R_i} \Pr(E^c \mid A' \in [A]). \tag{4.5}$$

Define now, for every $c \in \Sigma^n$, the set $B_c = \{i \in \{1, \ldots, k\} \mid c \in R_i\}$. By symmetry, $|R_i| = |R_j|$ for all $i, j \in \{1, \ldots, k\}$; and $|B_c| = |B_{c'}|$ for all $c, c' \in \Sigma^n$. Thus, Equation 4.5 transforms into:

$$r \sum_{i=1}^{k} \Pr(E^x \mid A' \in [A_i]) = b \sum_{c \in \{0,1\}^n} \Pr(E^c \mid A' \in [A]) \tag{4.6}$$

where $r = |R_1|$ and $b = |B_{0\ldots0}|$. Hence, from Equations 4.3 and 4.6 we obtain:

$$\Pr(E^x) = \frac{b}{r \cdot k} \sum_{c \in \Sigma^n} \Pr(E^c \mid A' \in [A]). \tag{4.7}$$

But, given that $[A] = \{A\}^{\mathcal{S}}$, from Equation 4.2 it follows:

$$\Pr(E^c \mid A' \in [A]) = \frac{1}{|\Sigma|^n |\Gamma|^n} \sum_{z \in \Sigma^n} |\Gamma|^{|meet_A(c,z)|}. \tag{4.8}$$

Therefore, by using Equations 4.7 and 4.8 we obtain:

$$\Pr(E^x) = \frac{b}{r \cdot k} \cdot \frac{c}{|\Sigma|^n |\Gamma|^n} \sum_{c,z \in \Sigma^n} |\Gamma|^{|meet_A(c,z)|}. \tag{4.9}$$

Observe that the right-hand side of Equation 4.9 does not depend on x. This means that $mafia(Proto) = \Pr(E^x)$ for all $x \in \Sigma^n$. Furthermore, $\sum_{i=1}^{k} |R_i| = \sum_{c \in \Sigma^n} |B_c|$ which gives us $r \cdot k = |\Sigma|^n \cdot b$, which together with Equation 4.9 give us the expected formula. □

The closed formulas from Lemmas 4.1 and 4.2 will be used in the next section to formalize and solve the security and space complexity trade-off optimality problem, within the class of layered and random-labeled lookup-based protocols. Recall that a random-labeled protocol is a protocol that is closed under $\mathcal{S}$.

4.2 Security and Size Trade-Off

We are interested in the trade-off between security in terms of mafia fraud and space complexity of the protocol. As we motivated in Chapter 1, this choice of security and performance indicators follows from two reasons: (1) mafia fraud is the fundamental attack that distance-bounding protocol must defend against, and (2) distance-bounding protocols

are mainly run on ubiquitous, resource-constrained devices such as RFID smart cards. On the one hand, security in terms of mafia fraud is determined by the protocol's resistance to pre-ask attacks. Space complexity, on the other hand, is interpreted as the largest amount of memory the device running the protocol must be able to allocate.

The memory of a protocol is determined as a function in the number n of fast phase rounds and the protocol-specific parameters. For example, the memory of the HK protocol is $f(n) = 2n$ which is function with the single argument n because the HK protocol has no further parameters. A more illustrative example is the u-uniform (recall Definition 3.11), whose memory is[1]:

$$\begin{aligned} f_1(u,n) &= \sum_{i=1}^{n} 2^{\min(u,i)} = \sum_{i=1}^{u} 2^i + \sum_{i=u+1}^{n} 2^u \\ &= 2^{u+1} - 2 + (n-u)2^u = (n-u+2)2^u - 2. \end{aligned}$$

In line with this computation, let us conduct a more general analysis. Let $h \in \{1, \ldots, 2^n\}$ be a tight upper bound on the number of states of any layer of any automaton of a given binary layered lookup-based protocol. Therefore, if no further properties on its automata are assumed, the memory of the given protocol is:

$$\begin{aligned} f_2(h,n) &= \sum_{i=1}^{n} \min(h, 2^i) = \sum_{i=1}^{\lfloor \log_2 h \rfloor} 2^i + \sum_{i=\lfloor \log_2 h \rfloor + 1}^{n} h \\ &= 2^{\lfloor \log_2 h \rfloor + 1} - 2 + (n - \lfloor \log_2 h \rfloor)\, h. \end{aligned} \tag{4.10}$$

For example, $h = 2^n$ for the Tree protocol [AT09] and therefore, from Equation 4.10 it follows that the Tree protocol requires $f_2(2^n, n) = 2^{n+1} - 2$ bits of memory, which is consistent with the representation given in Proposition 3.2. Hence, following the computation as in Equation 4.10 above, we will use h as the precise space measure, which we call the *size*.

Definition 4.5 (Size). The *size* of a layered automaton $A \in \mathbb{A}_{\Sigma,\Gamma}$, denoted by $size(A)$, is the number of states of the largest layer of A. Furthermore, the size of a protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$, denoted by $size(Proto)$, is equal to $\max_{A \in Proto}\{size(A)\}$.

The security and size trade-off optimality problem is therefore defined as follows.

Definition 4.6 (Security and Size Trade-Off Optimality Problem). Given $h \in \{1, \ldots, |\Sigma|^n\}$, find a protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ such that $mafia\,(Proto) \leq mafia\,(Proto')$ for every protocol $Proto' \subseteq \mathbb{A}_{\Sigma,\Gamma}$ that is layered, closed under $\mathcal{S}$, and whose size is not larger than h.

In Theorem 4.1 below we show a protocol transformation, which uses the closure with respect to $\mathcal{L}$, that results in a more secure protocol (or at least not less secure) protocol in terms of mafia fraud attacks based on the pre-ask strategy. We demonstrate that for every layered and random-labeled protocol (recall that a random-labeled property is equivalent to being closed under $\mathcal{S}$), there exists an automata in the protocol whose closure with respect to $\mathcal{L}$ is at least as resistant to pre-ask attacks as the original protocol. This is a

[1] Note that this value is smaller than $2^u n$ given in Figure 3.3 because the construction in the figure is not memory-wise optimal.

preliminary step towards solving the trade-off optimality problem of Definition 4.6, later on in Section 4.3.

Theorem 4.1. Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a layered protocol that is closed under $\mathcal{S}$, then:

$$\exists A \in Proto.\ mafia\left(\{A\}^{\mathcal{L}}\right) \leq mafia(Proto).$$

Proof. Let $A_1, \ldots, A_k \in Proto$ such that $[A_1], \ldots, [A_k]$ are the equivalence classes of $Proto$ with respect to $\mathcal{S}$. Let A be any of the automata from $Proto$ such that:

$$mafia\left(\{A\}^{\mathcal{L}}\right) = \min_{i\in\{1,\ldots,k\}} \left\{mafia\left(\{A_i\}^{\mathcal{L}}\right)\right\}. \tag{4.11}$$

We define, as in Proposition 4.1, for every $x \in \Sigma^n$, the event E^x that, for a random $c \in \Sigma^n$ and a random $B = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$, both (B, x, y) and (B, c, y) are correct executions of $Proto$, for some $y \in \Gamma^n$. From the law of total probability it follows that, for every $x \in \Sigma^n$:

$$\Pr(E^x) = \sum_{i=1}^{k} \Pr(E^x \mid B \in [A_i]) \Pr(B \in [A_i]). \tag{4.12}$$

From Proposition 4.1 we deduce that:

$$mafia(Proto) \geq \frac{1}{|\Sigma|^n} \sum_{x\in\Sigma^n} \Pr(E^x). \tag{4.13}$$

Hence, by using Equations 4.12 and 4.13 and inverting the order of the sums we obtain:

$$mafia(Proto) \geq \sum_{i=1}^{k} \left(\frac{1}{|\Sigma|^n} \sum_{x\in\Sigma^n} \Pr\left(E^x \mid B \in [A_i]\right) \right) \Pr(B \in [A_i]). \tag{4.14}$$

On the other hand, from Lemmas 4.1 and 4.2, and given that $[A_i] = \{A_i\}^{\mathcal{S}}$, for all $i \in \{1, \ldots, k\}$ we have:

$$\frac{1}{|\Sigma|^n} \sum_{x\in\Sigma^n} \Pr\left(E^x \mid B \in [A_i]\right) = mafia\left(\{A_i\}^{\mathcal{L}}\right)$$

Further, due to Equation 4.11 and given that $\sum_{i=1}^{k} \Pr(B \in [A_i]) = 1$, the inequality in Equation 4.14 transforms into $mafia(Proto) \geq mafia\left(\{A\}^{\mathcal{L}}\right)$ which completes the proof. □

Notice that consistency with respect to $\mathcal{L}$ imposes a structural property on a protocol: all automata of the protocol are equal if the labels on edges and states are ignored. Thus, Theorem 4.1 rules out, for example, protocol composition as a technique to obtain an optimal protocol, where a protocol composition is simply the union of the sets of automata defining the protocols. For instance, the union of HK [HK05] and the Tree [AT09] protocols does not result in an optimal protocol.

We have proven that the protocol transformation consisting of the application of Theorem 4.1 results in a protocol that is more secure than (or at least as secure as) the original one. In addition, such protocol transformation either preserves or simplifies the size of the original protocol. Hence, Theorem 4.1 reduces the search space to the subclass of protocols that are consistent and closed with respect to $\mathcal{L}$, in order to solve the security and size trade-off optimality problem. In the next section we address such optimality problem by providing a protocol solution.

4.3 The Modular Protocol

In mathematics, usually the optimal (minimum or maximum) value of a function on a number of variables takes place when such variables differ to each other in as least as possible. For example, as per the inequality of arithmetic and geometric means[2], $\left(\frac{s}{k}\right)^k$ is the maximum value of the product of k real-positive numbers whose sum is s. Such maximum takes place when all numbers are the same, i.e. equal to $\frac{s}{k}$.

Inspired by this, we define a property on automata that distributes the inputs over the states as evenly as possible. Intuitively, the more evenly the inputs are distributed over the states, the less probable (overall) it becomes for the adversary to reach a state visited during the pre-ask session.

In order to define an automata $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ that distribute evenly the input sequences over the states, we consider the transition function such that it minimizes $\sum_{q \in Q_i} (reach(q))^2$ for every $i \in \{1, \ldots, n\}$ and Q_i are the layers of A and $reach \colon Q \to \mathbb{N}$ gives the number of sequences that reach the argument state. To minimize these sums, we utilize remainder (or modular) operations on integer numbers. Later we will prove that the sums are related to the security of the protocol and that indeed the modular distribution property of δ indeed minimizes these sums.

Definition 4.7 (h-modular Automaton)**.** Let $h \in \{1, \ldots, |\Sigma|^n\}$, the h-modular automaton $A \in \mathbb{A}_{\Sigma,\Gamma}$ is a layered automaton such that:

$$\forall i \in \{1, \ldots, n\}, x, y \in \Sigma^i.$$
$$state_A(x) = state_A(y) \iff dec(x) \equiv dec(y) \pmod{h},$$

where $dec \colon \Sigma^* \to \mathbb{N}$ is defined by:

$$dec(c_1 \ldots c_n) = \sum_{j=0}^{n-1} |\Sigma|^j c_{n-j},$$

Note that $dec(x)$ refers to the decimal number represented by x in base $|\Sigma|$. The h-modular protocol is a closure with respect to $\mathcal{L}$ of a singleton set composed of the h-modular automaton.

Definition 4.8 (h-modular Protocol)**.** Let $h \in \{1, \ldots, |\Sigma|^n\}$, the h-modular protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ is defined as $Proto = \{A\}^{\mathcal{L}}$ where $A \in \mathbb{A}_{\Sigma,\Gamma}$ is the h-modular automaton.

The HK and the Tree protocols are subsets of the binary 2-modular and 2^n-modular protocols, respectively. From this point on we will often use the term "the Modular protocol" to refer to any h-modular protocol, for $h \in \{1, \ldots, |\Sigma|^n\}$. The reason for this simplification is that, even though there are $|\Sigma|^n$ of those protocols (if viewed as sets of automata), in a high-level, cryptographic context they all can be seen as one *instance* of the same protocol, and the instance is given by the choice of the security parameter h.

Observe that each automata of the h-modular protocol has the *largest girth* amongst all layered automata with size no longer than h. The girth is the shortest cycle contained in an automata if viewed as an undirected graph.

[2] See https://en.wikipedia.org/wiki/Inequality_of_arithmetic_and_geometric_means

4.3.1 Optimality Proof of the Modular Protocol

We observe that mafia fraud resistance in layered protocols strongly relates to the number of rounds where the adversary and verifier challenge sequence meet, as the adversary knows the correct answer in those rounds. This notion was introduced as the *set of meeting points* in Definition 4.4.

Given a protocol *Proto* that is consistent and closed with respect to $\mathcal{L}$ and $j \in \{1, \ldots, n\}$, the probability that $j \in meet_A(x, c)$, for two random sequences $x, c \in \Sigma^j$ and a random automaton $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell) \in Proto$ is equal to the number of pairs of j-input sequences that meet at any state in the j-th layer of A, divided by $|\Sigma|^{2j}$. Moreover, such number of pairs can be computed as $\sum_{i=1}^{|Q_j|} a_i^2$ where a_i is the number of sequences that reach the i-th state in that layer Q_j. Next, we provide a lower bound on this sum which, by extension, imposes a lower bound on the probability of success of a pre-ask attack.

Lemma 4.3. Let $h, j, p \in \mathbb{N}$ be three positive integers. Let $a_1 \geq a_2 \geq \cdots \geq a_h$ be non-negative integer numbers satisfying that $\sum_{i=1}^{h} a_i = p^j$. Then:

$$\sum_{i=1}^{h} a_i^2 \geq \omega(h, j, p),$$

where $\omega(h, j, p) = ha^2 + 2ar + r$, and $a = \left\lfloor p^j/h \right\rfloor$ and $r = p^j - ah$. The equality holds if $a_i \in \{a, a+1\}$ for $i \leq \min(h, p^j)$ and $a_i = 0$, otherwise.

Proof. Let $k \leq h$ be a positive integer number such that $a_i > 0$ if $i \leq k$ and $a_i = 0$, otherwise. Thus $\sum_{i=1}^{h} a_i^2 = \sum_{i=1}^{k} a_i^2$.

Let's assume $h \geq p^j$, then $\sum_{i=1}^{h} a_i^2 \geq \sum_{i=1}^{h} a_i = p^j = \omega(h, j, p)$. The equality holds if $a_i = 1, \forall i \leq k$ which implies that $k = p^j$.

Consider now $h < p^j$. Observe that $a_1 \geq 2$, otherwise $a_1 = \cdots = a_k = 1$ and, consequently, $k = p^j$ but $k \leq h$, which is a contradiction. Now, observe that $\sum_{i=1}^{k} a_i^2 > 1 + (a_1 - 1)^2 + \sum_{i=2}^{k} a_i^2$. Thus, there exists a permutation $(b_1, b_2, \ldots, b_{k+1})$ of $(1, a_1 - 1, a_2, \ldots, a_k)$ such that $b_1 \geq b_2 \geq \cdots \geq b_{k+1} > 0$ and $\sum_{i=1}^{k} a_i^2 > \sum_{i=1}^{k+1} b_i^2$. Hence, as we are minimizing, we can successively apply the same process $h - k$ times, until reaching $c_1 \geq c_2 \geq \cdots \geq c_h > 0$ such that $\sum_{i=1}^{h} c_i = p^j$ and $\sum_{i=1}^{h} c_i^2 \leq \sum_{i=1}^{k} a_i^2$.

Now, let $d = c_1 - c_h$ and suppose $d > 1$. Let $e_1 = c_1 - d + 1$, $e_h = c_h + d - 1$ and $e_i = c_i$ for all $i \in \{2, \ldots, h-1\}$. Notice that $\sum_{i=1}^{h} e_i = p^j$, $e_1 > 0$ and $e_h > 0$. It is easy to verify that $e_1^2 + e_h^2 < c_1^2 + c_h^2$, which implies that $\sum_{i=1}^{h} e_i^2 < \sum_{i=1}^{h} c_i^2$. Again, as we are looking for the minimum, we can successively apply the same process until reaching $t_1 \geq t_2 \geq \cdots \geq t_h > 0$ such that $t_1 - t_h \leq 1$, $\sum_{i=1}^{h} t_i = p^j$ and $\sum_{i=1}^{h} t_i^2 \leq \sum_{i=1}^{h} c_i^2$.

Hence, given that $t_1 - t_h \leq 1$ we derive that $t_i \in \{t_h, t_h + 1\}$ for every $i \in \{1, \ldots, h\}$. This gives $t_i = t_h + 1$ if $i \leq r$ and $t_i = t_h$ otherwise, where $r = p^j - h\lfloor p^j/h \rfloor$ (i.e. $r \equiv p^j \pmod{h}$). Finally, $\sum_{i=1}^{h} t_i^2 = \omega(h, j, p)$. □

For the remaining of this chapter, whenever we use ω we refer to the function defined in Lemma 4.3. The next corollary is a consequence of Lemma 4.2. It will serve to prove the main result of this chapter later on in Theorem 4.2.

Corollary 4.1. Let $A \in \mathbb{A}_{\Sigma,\Gamma}$ be a layered automaton and let $B_J = \{(x, y) \in \Sigma^{2n} \mid J \subseteq meet_A(x, y)\}$ for all $J \subseteq \{0, \ldots, n\}$. Let $W_0 = |\Sigma|^{2n}$ and $W_k = \sum_{J \subseteq \{0,\ldots,n\}, |J|=k} |B_J|$ for

all $k > 0$. Then:

$$mafia\left(\{A\}^{\mathcal{L}}\right) = \frac{1}{|\Sigma|^{2n}|\Gamma|^n} \sum_{i=0}^{n} W_i \left(|\Gamma| - 1\right)^i .$$

Proof. Lemma 4.2 gives us:

$$mafia\left(\{A\}^{\mathcal{L}}\right) = \frac{1}{|\Sigma|^{2n}|\Gamma|^n} \sum_{x,y \in \Sigma^n} |\Gamma|^{|meet_A(x,y)|} = \frac{1}{|\Sigma|^{2n}|\Gamma|^n} \sum_{k=0}^{n} |\Gamma|^k N_k$$

where N_k is the number of pairs $(x, y) \in \Sigma^{2n}$ such that they collide exactly k times when traversing through A, i.e. $N_k = \left|\left\{(x, y) \in \Sigma^{2n} \mid |meet_A(x, y)| = k\right\}\right|$. Hence, from the combinatorial inclusion-exclusion principle we have that $N_k = \sum_{i=k}^{n} (-1)^{i+k} \binom{i}{k} W_i$ for all $k \in \{0, \ldots, n\}$. This gives us the expected formula. □

In the next lemma we explicitly link the structural properties of the Modular protocol with the function ω defined in Lemma 4.3. This is an important result for our optimality proof as ω determines the lower bound on the overall collisions of two sequences on a given automaton.

Lemma 4.4. Let $h \in \{1, \ldots, |\Sigma|^n\}$ and let $A = (\Sigma, \Gamma, Q, q_0, \delta, \ell)$ be the h-modular automaton. Let $\{Q_0, \ldots, Q_n\}$ be the partition of Q representing the layers of A. Then, for all $i, j \in \{0, \ldots, n\}$ with $i < j$ and all $q \in Q_i$, it holds that:

$$\omega(h, j - i, |\Sigma|) = \left|\left\{(x, y) \in \Sigma^{2(j-i)} \mid state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(x) = state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(y)\right\}\right| .$$

Proof. Let T be the set containing all tuples $(i, j, q) \in \mathbb{N} \times \mathbb{N} \times Q$ such that $0 \leq i < j \leq n$ and $q \in Q_i$. Consider now the function $f \colon T \to \mathbb{N}$ defined by:

$$f(i, j, q) = \left|\left\{(x, y) \in \Sigma^{2(j-i)} \mid state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(x) = state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(y)\right\}\right| .$$

Our goal is to prove that $f(i, j, q) = \omega(h, j - i, |\Sigma|)$ for all $(i, j, q) \in T$. In effect, since A is the the h-modular automaton, it follows that, for all $(i, j, q) \in T$ and all $x, y \in \Sigma^{j-i}$:

$$state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(x) = state_{(\Sigma,\Gamma,Q,q,\delta,\ell)}(y) \iff dec(x) \equiv dec(y) \pmod{h}. \tag{4.15}$$

Consider now a tuple $(i, j, q) \in T$ and, for every $r \in \{1, \ldots, h\}$, let a_r be the number of $(j - i)$-length input sequences such that, starting from q, reach the r-th state in Q_j ($a_r = 0$ means that either such state does not exist or is not reachable from q). Therefore, $f(i, j, q) = \sum_{r=1}^{h} a_r^2$. Hence, because of Equation 4.15, we deduce that there exists a permutation $(b_1, b_2, \ldots, b_h)$ of $(a_1, a_2, \ldots, a_h)$ such that $b_1 \geq b_2 \geq \cdots \geq b_h \geq 0$ where $b_r = 0$ for $r > k = \min(h, |\Sigma|^{j-i})$ and $b_1 - b_k \leq 1$. Hence, from Lemma 4.3 we derive that $\sum_{r=1}^{h} b_r^2 = \omega(h, j - i, |\Sigma|)$, which concludes the proof. □

As mentioned before, the automata of the Modular protocol distribute as evenly as possible the inputs over the states. This property makes such protocol to perform well when facing mafia fraud. Furthermore, we prove in Theorem 4.2 next that the Modular protocol is more secure than (or as secure as) any layered protocol consistent and closed with respect to $\mathcal{L}$, as long as the latter has size no larger than the former.

Theorem 4.2. Let $A \in \mathbb{A}_{\Sigma,\Gamma}$ be a layered automata and $A' \in \mathbb{A}_{\Sigma,\Gamma}$ be the h-modular automaton with $h \geq size(A)$. Then:

$$mafia\left(\{A'\}^{\mathcal{L}}\right) \leq mafia\left(\{A\}^{\mathcal{L}}\right) .$$

Proof. For every $J \subseteq \{0,\ldots,n\}$ consider the sets $B_J = \{(x,y) \in \Sigma^{2n} \mid J \subseteq meet_A(x,y)\}$ and $B'_J = \{(x,y) \in \Sigma^{2n} \mid J \subseteq meet_{A'}(x,y)\}$. From Corollary 4.1 we derive that it is sufficient to prove that $|B_J| \geq |B'_J|$ for all $J \subseteq \{0,\ldots,n\}$.

Let $j_0 = 0$, $J = \{j_1,\ldots,j_k\} \subseteq \{0,\ldots,n\}$ with $0 < j_1 < j_2 < \cdots < j_k$ and $G \in \mathbb{N}$ defined as follows:

$$G = \prod_{t=1}^{k} \omega(h, j_t - j_{t-1}, |\Sigma|), \tag{4.16}$$

where ω is defined as in Lemma 4.3. Furthermore, for every $t \in \{1,\ldots,k\}$ consider $D(j_{t-1}, j_t)$ as the *minimum* number of pairs of $(j_t - j_{t-1})$-length input sequences that start from the same state in layer j_{t-1} and reach the same state in the layer j_t. Then:

$$|B_J| \geq |\Gamma|^{2n-2j_k} \prod_{t=1}^{k} D(j_{t-1}, j_t). \tag{4.17}$$

From Lemma 4.3 we derive that $D(j_{t-1}, j_t) \geq \omega(h, j_t - t_{t-1}, |\Sigma|)$ for all $t \in \{1,\ldots,k\}$. By using this and Equations 4.16 and 4.17 we obtain:

$$|B_J| \geq |\Gamma|^{2n-2j_k} \prod_{t=1}^{k} \omega(h, j_t - j_{t-1}, |\Sigma|) = |\Gamma|^{2n-2j_k} \times G. \tag{4.18}$$

Therefore, from Lemma 4.4 we have that $|B'_J| = |\Gamma|^{2n-2j_k} \times G$. Finally, by using this in Equation 4.18 we conclude that $|B_J| \geq |B'_J|$. □

With this last result along with the statement of the end of Section 4.2, we can conclude that, amongst all layered and random-labeled protocols with size of at most h, the h-modular protocol is optimal in terms of resistance to pre-ask attacks. This means that, the h-modular protocol is a solution for the security and size trade-off optimality problem (Definition 4.6). This is the main result of the current chapter, together with the construction of the h-modular protocol that we propose later on in Section 4.3.2.

Theorem 4.3 (h-modular is Optimal)**.** Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a layered protocol that is closed under $\mathcal{S}$. Let $M \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be the h-modular protocol with $h \geq size(Proto)$. Then:

$$mafia(M) \leq mafia(Proto) .$$

Proof. The proof follows straightforwardly from Theorems 4.1 and Theorem 4.2. □

In Theorem 3.1 we proved that the Tree protocol is the optimal binary lookup-based protocol in terms of resistance to pre-ask attacks. However, the protocol requires an exponential amount of memory, which makes it infeasible in practice. Motivated by this, we derive a straightforward consequence of Theorem 4.3: a protocol that is layered, closed under $\mathcal{S}$, and optimal with no size restrictions does have exponential size in the number n of round-trip time rounds.

Corollary 4.2. Let $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ be a layered protocol that is closed under $\mathcal{S}$ such that there is no layered protocol $Proto' \subseteq \mathbb{A}_{\Sigma,\Gamma}$ closed under $\mathcal{S}$ as well, with $mafia(Proto') < mafia(Proto)$. Then:

$$size(Proto) = |\Sigma|^n.$$

Proof. Let $h = size(Proto)$ and let M be the h-modular protocol and let M' be the $|\Sigma|^n$-modular protocol. Let's assume that $h < |\Sigma|^n$. Hence, from Theorem 4.3 it follows that no layered protocol $Proto' \subseteq \mathbb{A}_{\Sigma,\Gamma}$ exists such that $mafia(Proto') < mafia(M)$. But, this is a contradiction because $mafia(M') < mafia(M)$ due to Lemma 4.4 and the composition of $\omega(...)$ (recall this function from Lemma 4.3). □

We remark that the size of a protocol $Proto \subseteq \mathbb{A}_{\Sigma,\Gamma}$ is never greater than $|\Sigma|^n$. This trivially follows from the fact that the maximum number of possible reachable state on a given automaton is upper-bounded by $|\Sigma|^n$, being the number of different sequences of length n.

4.3.2 Constructing the Modular Protocol

In this section we provide a description of the binary h-modular protocol, for any $h \in \{1, \ldots, 2^n\}$, in standard cryptographic notation for security protocols. The protocol is illustrated in Figure 4.1 and consists of the following phases:

1. *Initialization:* The parties agree on protocol parameters such as: a secret and shared key x, the length m of the nonces, the number of rounds n, the size $h \in \{1, \ldots, 2^n\}$ of the automata, a pseudo-random function f, and a threshold Δt_{max} for the round-trip times.

2. *Slow phase:* In the slow phase, the prover and verifier exchange nonces N_V and N_P. The output of the pseudo-random function $f(x, N_V, N_P)$ is used to create h sequences $T^0, \ldots, T^{h-1}$ of n random bits each (numbered from 1 to n, i.e. $T^j = T^j_1 \cdots T^j_n$) and h sequences $D^0, \ldots, D^{h-1}$ of $n-1$ random bits each (numbered from 1 to $n-1$, i.e. $D^j = D^j_1 \cdots D^j_{n-1}$). The initial state q_0 is set to 0.

3. *Fast phase:* This phase is composed of n rounds numbered from 1 to n. At the i-th round, the verifier sends a challenge bit c_i to the prover. The prover moves from the q_{i-1}-th state in layer $i-1$ to the q_i-th state in layer i such that $q_i \in \{0, \ldots, h-1\}$ and $q_i \equiv 2q_{i-1} + c_i \oplus D^{q_{i-1}}_i \pmod h$ and replies with the bit $T^{q_i}_i$.

4. *Verification:* The protocol succeeds if and only if all round-trip times are below the pre-defined threshold Δt_{max} and all responses are correct.

4.4 Comparative Analysis

This section intends to compare the proposed Modular protocol with several state-of-the-art distance-bounding protocols, including protocols that are not lookup-based. Our hypothesis is that the the Modular protocol offers interesting features that are not offered by other distance-bounding protocols, such as a good trade-off between mafia fraud resistance

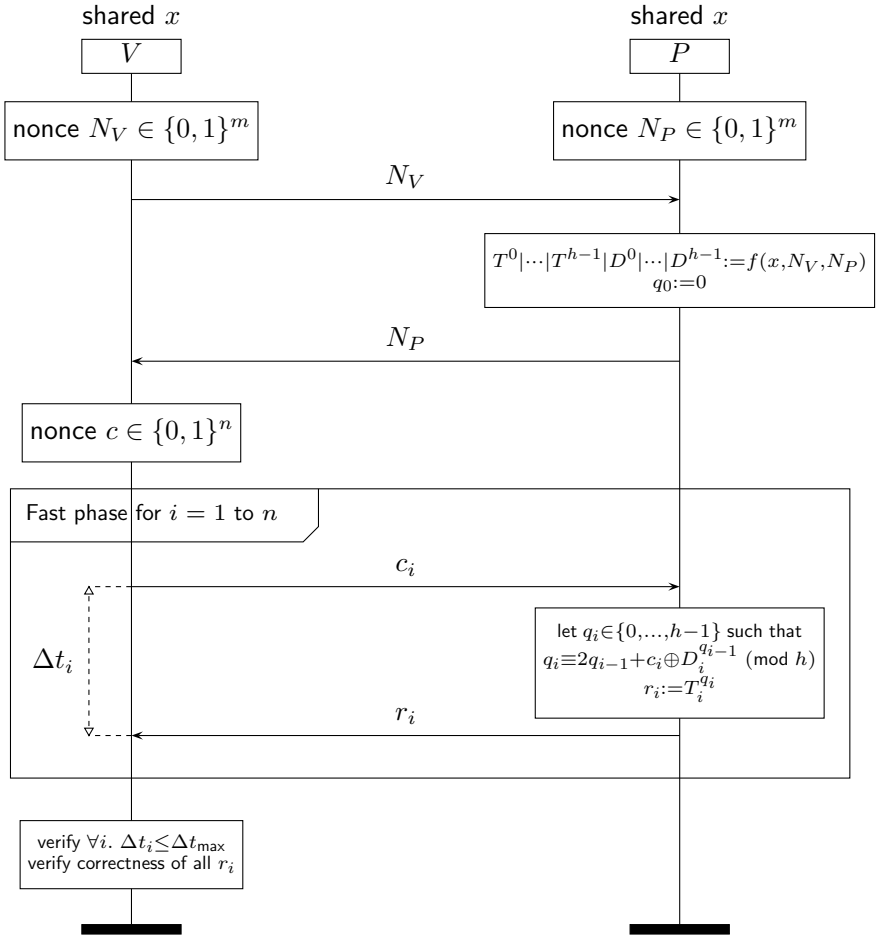


Figure 4.1 The binary h-modular protocol. This construction takes $(2n-1)h$ bits of memory.

and memory usage. To validate this hypothesis we consider the methodology proposed in [AMT15], which we describe next.

4.4.1 Framework of Reference

The software-supported[3] comparison methodology [AMT15] determines the best distance-bounding protocol(s) according to a set of criteria. Each criterion is defined by a triple $(D, <, \sim)$, where D is an attribute domain with total order $<$ and approximate equality $\sim$. For example, *memory* is a criterion with domain $\mathbb{N}$ representing the minimum size of the volatile memory (in bits) used by the prover during the protocol execution. The total order for the memory criterion is the usual total order for natural numbers, and the approximate equality $\sim$ is defined by $x \sim y \iff |x - y| < 1024$. That is to say, two memory attribute values x and y are considered similar if they differ in less than a kilobit.

The authors in [AMT15] highlight eight criteria that have been used frequently in the literature: number of bits exchanged during the RTT-measurement phase (e), probability of success of mafia (p_m), distance (p_d) and terrorist (p_t) fraud attacks, number of bits

[3] Available at https://github.com/rolandotr/db_comparison

(b) of the fast phase communication channel, number of cryptographic operations (c), memory (m), and whether the protocol has a prover-active final phase or not (f). For every $i \in \{e, p_m, p_d, p_t, b, c, m, f\}$, we use $(D_i, <_i, \sim_i)$ to refer to the triple corresponding to the criteria i. We kindly refer the reader to [AMT15] for intuitions and formal definitions of these attributes.

There exist many parameters that influence a protocol specification, such as number of rounds and size of the cryptographic keys. We thus use $\mathit{Inst}\,(\mathit{Proto})$, for a given protocol Proto, to denote the set of all possible parametrizations of Proto. If $p \in \mathit{Inst}\,(\mathit{Proto})$ then we say that p is a *protocol instance* or an instantiation of Proto. As in [AMT15], we assume that any protocol instance can be mapped to a value within the attribute space $\Omega = D_e \times D_{p_m} \times D_{p_d} \times D_{p_t} \times D_b \times D_c \times D_m \times D_f$. Such a mapping is represented by the total function $\mathit{Values}\colon \bigcup_{\mathit{Proto} \in \mathbb{P}} \mathit{Inst}\,(\mathit{Proto}) \to \Omega$ where $\mathbb{P}$ is the universe of all protocol specifications.

Intuitively, a protocol instance p_1 outperforms another protocol instance p_2 if it performs better in one criterion and better or equal in the others. This is formalized by the *dominant relation* $\prec \subseteq \Omega \times \Omega$ defined by:

$$x \prec y \iff (\forall i \in I.\ x_i < y_i \vee x_i \sim y_i) \wedge (\exists i \in I.\ x_i < y_i)$$

where $I = \{e, p_m, p_d, p_t, b, c, m, f\}$ and x_i denotes the element of x corresponding to the attribute domain D_i. The protocol p is said to be *dominated* by p' if $\mathit{Values}(p) \prec \mathit{Values}(p')$.

A dominant relation is typically used in multi-criteria decision-making to determine the largest subset of non-dominated objects, i.e., those that are not outperformed (dominated) by any other object. Below we precisely define the set of protocol instances to be compared, and show those protocol instances that turned out to be non-dominated.

4.4.2 Experimental Setting

For our experiments we kept a similar setting to that of [AMT15]. We considered the eight previously described criteria. The differences in our experiments with respect to [AMT15] are twofold:

- We reduced the number of rounds from 256 to 128 because the results do not change for $n > 128$, as shown in [AMT15].
- We added the proposed Modular protocol and its corresponding instances.

Given a protocol Proto, we use Proto-$\{p_1, p_2, \ldots, p_k\}$ to refer to the instance of Proto with parameter values $\{p_1, p_2, \ldots, p_k\}$. For example, BC-$\{12\}$ stands to the BC protocol with $n = 12$ rounds, while Tree-$\{28, 7\}$ represents the Tree protocol with trees of depth $l = 7$ and $n = 28$ rounds. The full list of protocols and parameters can be found in Table 4.1. In this table, an instance of the proposed Modular protocol is denoted by Modular-$\{n, h\}$, where n is again the number of rounds and h the size of the protocol.

As we do not provide the analysis of distance and terrorist fraud attacks against the Modular protocol, we set the values $p_d = \frac{1}{2^{n+1}} + \frac{1}{2}\sqrt{\frac{1}{2^{2n}} - \frac{4}{2^n} + 4p_m}$ and $p_t = 1$. In the case of distance fraud, it corresponds to the upper bound provided in [TMA10].

Table 4.1 Protocol instances selected for the multi-criteria comparison experiments.

Protocol	Identifier	Parameter values
Brands and Chaum [BC93]	BC-$\{n\}$	$n \in \{1, \ldots, 128\}$
MAD [CBH03]	MAD-$\{n\}$	$n \in \{1, \ldots, 128\}$
Bussard and Bagga [BB05]	BB-$\{n\}$	$n \in \{1, \ldots, 128\}$
Hancke and Kuhn [HK05]	HK-$\{n\}$	$n \in \{1, \ldots, 128\}$
Munilla and Peinado [MP08]	MP-$\{n, p_f\}$	$n \in \{1, \ldots, 128\}$ $p_f \in \{0, 0.05, 0.1, \ldots, 0.95, 1\}$
SwissKnife [KAK+08]	SwissKnife-$\{n\}$	$n \in \{1, \ldots, 128\}$
Tree [AT09]	Tree-$\{n, l\}$	$n \in \{1, \ldots, 128\}$ $l \in \{1, 2, \ldots, 32\}$
Poulidor [TMA10]	Poulidor-$\{n\}$	$n \in \{1, \ldots, 128\}$
Rasmussen and Capkun [RC10]	RC-$\{n\}$	$n \in \{1, \ldots, 128\}$
Yum et al. [YKHL11]	YKHL-$\{n\}$	$n \in \{1, \ldots, 128\}$
Kim and Avoine [KA11]	KA-$\{n, p_d\}$	$n \in \{1, \ldots, 128\}$ $p_d \in \{0, 0.05, 0.1, \ldots, 0.95, 1\}$
SKI [BMV13a]	SKI-$\{n, t\}$	$n \in \{1, \ldots, 128\}$ $t \in \{2, 3, \ldots, 32\}$
TMA [TMA14]	TMA-$\{n\}$	$n \in \{1, \ldots, 128\}$
Modular	Modular-$\{n, h\}$	$n \in \{1, \ldots, 128\}$ $h \in \{16, 32, 64, 128, 256\}$

4.4.3 Results

According to the above settings, a total of 15232 protocol instances were analyzed. We ran the decision-making approach proposed in [AMT15] six times. Each run corresponds to the analysis of a set of protocol instances that resist to mafia fraud with probability at least y, where $y \in \{2^{-16}, 2^{-32}, 2^{-48}, 2^{-64}, 2^{-80}, 2^{-96}\}$. The sets of protocol instances are denoted by I_y, for every y. The results are depicted in Table 4.2.

To review the comparison results, we refer to Figures 4.2 and 4.3, which depict charts with mafia and distance fraud success probabilities for a number of protocol instances. We have selected representative protocol instances based on the most frequent protocol parameters (except for n) as per Table 4.2. The selected protocol instances are KA-$\{n, 0.95\}$, BC-$\{n\}$, Tree-$\{n, 6\}$, TMA-$\{n\}$, Poulidor-$\{n\}$, SwissKnife-$\{n\}$, Modular-$\{n, 32\}$, and SKI-$\{n, 2\}$, for $n \in \{10, 20, \ldots, 120\}$. We provide next a brief discussion on the reasons for the non-dominance of the protocol instances of Table 4.2:

- The instances of the BC, BB, MAD, and RC protocols achieve the optimal resistance to both mafia and distance fraud amongst all instances of protocols with single-bit channel for the fast phase. In addition, the BC protocol outperforms BB, MAD, and RC protocols, hence excluding the instances of these three latter protocols from the non-dominated sets.

Table 4.2 Non-dominated protocol instances for the different sets I_y. This table only shows, for each protocol, the non-dominated protocol instance (if any) with least memory usage and fewer bits exchanged during the fast phase, in that order. The total number of non-dominated instances is given in the last column. Furthermore, every security value p has been scaled to $2^{\lceil \log_2 p \rceil}$, and the memory values m have been scaled to $\lfloor m/1024 \rfloor$ kilobits.

y	**Non-dominated instances in I_y**	**Attribute values**								**total**
		n	p_m	p_d	p_t	b	c	m	f	
2^{-16}	KA-{20, 0.85}	20	$2^{-16.0}$	$2^{-2.0}$	$2^{0.0}$	1	1	0	*false*	3
	BC-{16}	16	$2^{-16.0}$	$2^{-16.0}$	$2^{0.0}$	1	2	0	*true*	113
	Tree-{24, 6}	24	$2^{-16.0}$	$2^{-10.0}$	$2^{0.0}$	1	1	1	*false*	181
	TMA-{27}	27	$2^{-16.0}$	$2^{-16.0}$	$2^{0.0}$	1	1	0	*false*	1
	SwissKnife-{16}	16	$2^{-16.0}$	$2^{-6.0}$	$2^{-6.0}$	1	2	1	*true*	113
	Modular-{21, 16}	21	$2^{-16.0}$	$2^{-8.0}$	$2^{0.0}$	1	1	1	*false*	4
	SKI-{39, 2}	39	$2^{-16.0}$	$2^{-16.0}$	$2^{-39.0}$	2	1	1	*false*	90
2^{-32}	Poulidor-{42}	42	$2^{-32.0}$	$2^{-16.0}$	$2^{0.0}$	1	1	0	*false*	1
	KA-{37, 0.9}	37	$2^{-32.0}$	$2^{-2.0}$	$2^{0.0}$	1	1	0	*false*	2
	BC-{32}	32	$2^{-32.0}$	$2^{-32.0}$	$2^{0.0}$	1	2	0	*true*	97
	Tree-{48, 6}	48	$2^{-32.0}$	$2^{-21.0}$	$2^{0.0}$	1	1	1	*false*	156
	TMA-{53}	53	$2^{-32.0}$	$2^{-32.0}$	$2^{0.0}$	1	1	0	*false*	1
	SwissKnife-{32}	32	$2^{-32.0}$	$2^{-13.0}$	$2^{-13.0}$	1	2	1	*true*	97
	Modular-{41, 16}	41	$2^{-32.0}$	$2^{-16.0}$	$2^{0.0}$	1	1	1	*false*	4
	SKI-{78, 2}	78	$2^{-32.0}$	$2^{-32.0}$	$2^{-78.0}$	2	1	1	*false*	51
2^{-48}	Poulidor-{61}	61	$2^{-48.0}$	$2^{-25.0}$	$2^{0.0}$	1	1	0	*false*	1
	KA-{53, 0.95}	53	$2^{-48.0}$	$2^{-1.0}$	$2^{0.0}$	1	1	0	*false*	4
	BC-{48}	48	$2^{-48.0}$	$2^{-48.0}$	$2^{0.0}$	1	2	0	*true*	81
	Tree-{72, 6}	72	$2^{-48.0}$	$2^{-32.0}$	$2^{0.0}$	1	1	2	*false*	120
	TMA-{80}	80	$2^{-48.0}$	$2^{-48.0}$	$2^{0.0}$	1	1	0	*false*	1
	SwissKnife-{48}	48	$2^{-48.0}$	$2^{-19.0}$	$2^{-19.0}$	1	2	1	*true*	81
	Modular-{58, 32}	58	$2^{-48.0}$	$2^{-24.0}$	$2^{0.0}$	1	1	4	*false*	4
	SKI-{116, 2}	116	$2^{-48.0}$	$2^{-48.0}$	$2^{-116.0}$	2	1	1	*false*	13
2^{-64}	Poulidor-{78}	78	$2^{-64.0}$	$2^{-32.0}$	$2^{0.0}$	1	1	0	*false*	1
	KA-{70, 0.9}	70	$2^{-64.0}$	$2^{-2.0}$	$2^{0.0}$	1	1	0	*false*	4
	BC-{64}	64	$2^{-64.0}$	$2^{-64.0}$	$2^{0.0}$	1	2	0	*true*	65
	Tree-{96, 6}	96	$2^{-64.0}$	$2^{-43.0}$	$2^{0.0}$	1	1	2	*false*	83
	TMA-{106}	106	$2^{-64.0}$	$2^{-64.0}$	$2^{0.0}$	1	1	0	*false*	1
	SwissKnife-{64}	64	$2^{-64.0}$	$2^{-26.0}$	$2^{-26.0}$	1	2	1	*true*	65
	Modular-{77, 32}	77	$2^{-64.0}$	$2^{-32.0}$	$2^{0.0}$	1	1	5	*false*	4
	SKI-{110, 3}	110	$2^{-64.0}$	$2^{-45.0}$	$2^{-64.0}$	3	1	1	*false*	1
2^{-80}	Poulidor-{96}	96	$2^{-80.0}$	$2^{-41.0}$	$2^{0.0}$	1	1	0	*false*	1
	KA-{86, 0.95}	86	$2^{-80.0}$	$2^{-2.0}$	$2^{0.0}$	1	1	0	*false*	5
	BC-{80}	80	$2^{-80.0}$	$2^{-80.0}$	$2^{0.0}$	1	2	0	*true*	49
	Tree-{120, 6}	120	$2^{-80.0}$	$2^{-54.0}$	$2^{0.0}$	1	1	3	*false*	46
	Modular-{95, 32}	95	$2^{-80.0}$	$2^{-40.0}$	$2^{0.0}$	1	1	6	*false*	4
	SwissKnife-{80}	80	$2^{-80.0}$	$2^{-33.0}$	$2^{-33.0}$	1	2	1	*true*	49
	SKI-{118, 4}	118	$2^{-80.0}$	$2^{-48.0}$	$2^{-48.0}$	4	1	1	*false*	1
2^{-96}	Poulidor-{114}	114	$2^{-96.0}$	$2^{-49.0}$	$2^{0.0}$	1	1	1	*false*	1
	KA-{102, 0.95}	102	$2^{-96.0}$	$2^{-2.0}$	$2^{0.0}$	1	1	0	*false*	5
	BC-{96}	96	$2^{-96.0}$	$2^{-96.0}$	$2^{0.0}$	1	2	0	*true*	33
	Tree-{126, 14}	126	$2^{-99.0}$	$2^{-60.0}$	$2^{0.0}$	1	1	288	*false*	14
	Modular-{109, 64}	109	$2^{-96.0}$	$2^{-48.0}$	$2^{0.0}$	1	1	14	*false*	3
	SwissKnife-{96}	96	$2^{-96.0}$	$2^{-39.0}$	$2^{-39.0}$	1	2	1	*true*	33
	SKI-{124, 6}	124	$2^{-96.0}$	$2^{-51.0}$	$2^{-32.0}$	6	1	1	*false*	1

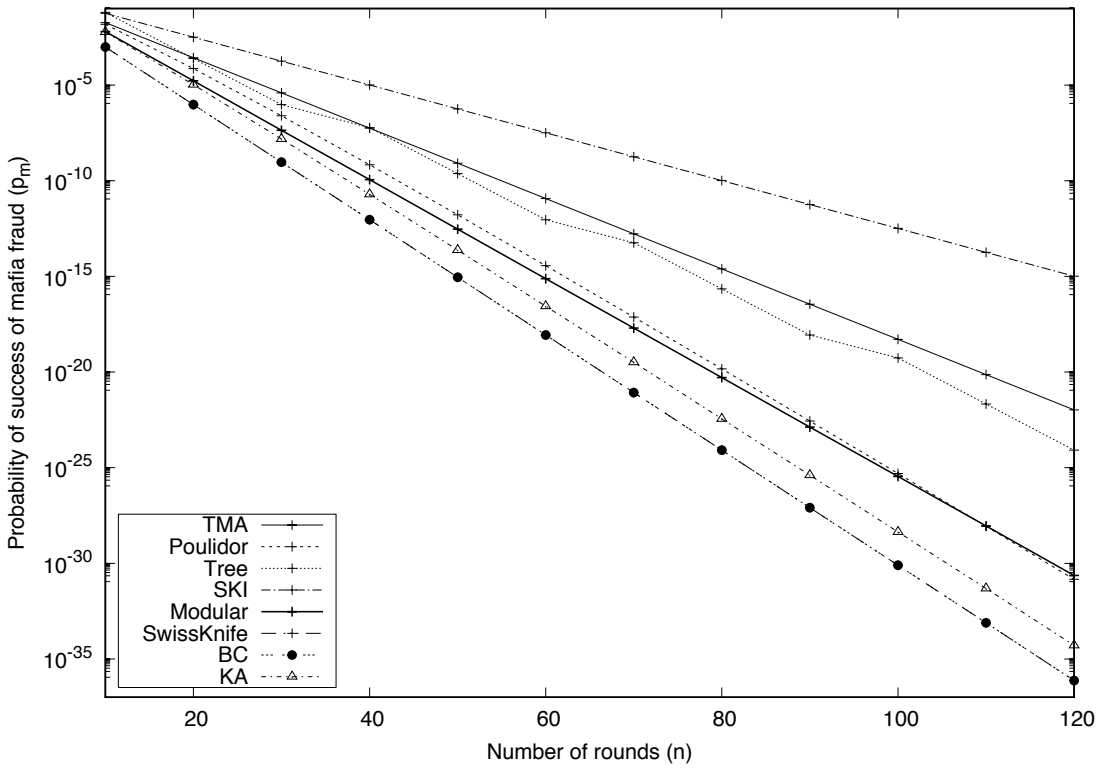


Figure 4.2 The probability of success of mafia fraud attacks against various protocol instances.

- The instances of the TMA protocol are non-dominated because they offer the best resistance to distance fraud amongst all protocol instances that have a single-bit fast phase channel and a single call to a cryptographic function (see Figure 4.3).

- The instances of the SwissKnife and SKI protocols are non-dominated because these are the only protocols that resist terrorist fraud attacks, and neither instance of them dominates the other. This is because the SKI protocol exchanges more bits than SwissKnife during the fast phase, and the SwissKnife protocol has a final, prover-active phase, which the SKI protocol does not.

- While the instances of the KA protocol do not perform well in terms of distance fraud (see Figure 4.3), they perform better in terms of mafia fraud (see Figure 4.2) than the instances of any other protocol with just one call to a cryptographic function.

- Overall, the instances of the Poulidor, Modular and Tree protocols perform similar security-wise. Furthermore, they offer the best resistance to mafia fraud (see Figure 4.2) amongst instances of protocols without a final, prover-active phase, except for the KA protocol. The KA protocol instances do not dominate the instances of any of these three protocols because KA protocol is the least resistant to distance fraud (see Figure 4.3).

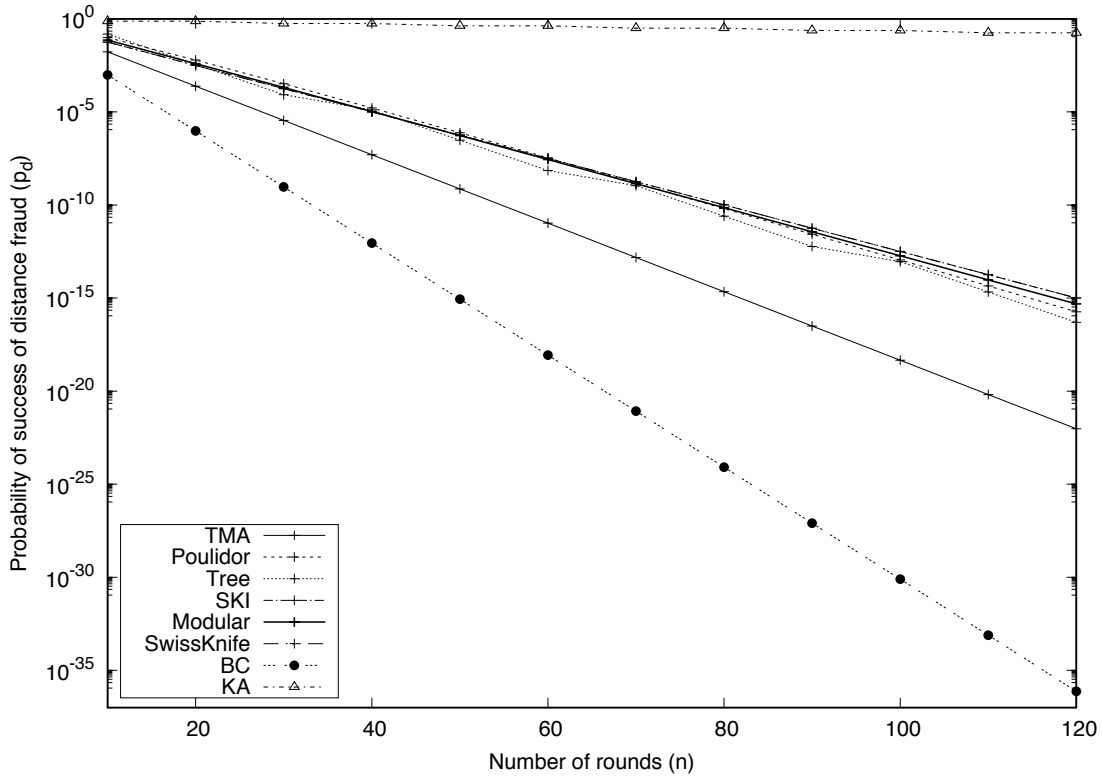


Figure 4.3 The probability of success of distance fraud attacks against various protocol instances.

None of the instances of the Poulidor, Modular and Tree protocols dominate any other because the Modular protocol performs better than the other two in terms of mafia fraud, the Poulidor protocol has less memory consumption (except for the set $I_{2^{-16}}$) than the other two, and the Tree protocol performs better than the other two in terms of distance fraud.

Table 4.2 confirms our hypothesis that the Modular protocol is relevant with respect to the most frequently used criteria in the literature. We observe that at least one protocol instance of the Modular protocol is non-dominated for every set I_y of Table 4.2. In general, we observe that the Modular protocol instances offer security levels comparable with various protocols that are not layered or even not lookup-based. All this, at a very low-cost in memory and computational complexity.

4.5 Conclusions

We have proposed the first lookup-based distance-bounding protocol that is optimal within a large class of protocols, in terms of resistance to mafia fraud for a given upper bound on the size of the protocol. A comprehensive set of experiments relying on decision-making theory, which included thirteen state-of-the-art protocols, indicate that the proposed protocol cannot be outperformed by other protocols, as it balances well security and memory

consumption. At the conceptual level, we showed that equivalence relations on automata shape a powerful tool for the analysis of distance-bounding protocols.

The proposed protocol seems to have connections with some works in further security fields. For example, the properties of bipartite and q-partite graphs, which relates to layered protocols, have been used to guarantee lower bounds on the code rate of error correcting codes [Tan81, KPP+04]. In addition, graphs with large girths, such as expander and Cayley graphs, have been studied in order to design provable-secure cryptographic hashes [Zém91].

Part II

Symbolic Analysis of Distance-Bounding Protocols

5

Causality-Based Secure Distance-Bounding

Existing frameworks for symbolic verification of distance-bounding protocols are based on events' timestamps and agents' location. These approaches differ from traditional symbolic verification frameworks because the latter deliver (in)security proofs by looking at the causality of the agents' actions.

In this chapter we introduce a causality-based property to verify distance-bounding protocols that discards the notions of continuous time and physical location. This constitutes a major step forward into computer-assisted verification of distance-bounding protocols as time and location are arguably hard to properly account for in established protocol verification tools.

Organization– In Section 5.1 we describe the Isabelle/HOL [NPW02]-supported symbolic model introduced by Basin et al. in [BCSS09, SSBC09] to verify distance-bounding protocols. Basin et al.'s model is based on the location of agents and the time at which the agents' events occur. In Section 5.2 we introduce a set of property to characterize a semantic protocol domain on which our results can be applied. In Section 5.3 we propose the causality-based secure distance-bounding property, which we prove equivalent to Basin et al.'s property in protocol models within the semantic domain. We give conclusive remarks of the chapter in Section 5.4.

5.1 A Model Based on Time and Location

This section describes the formalism of Basin et al. [BCSS09, SSBC09], which is the basis for the work developed in this chapter. The formalism employs logic theories to handle inductively-defined sets of traces that represent the protocol's executions. It considers execution traces that consist of a sequence of timestampted events, e.g. denoting the sending and reception of messages, where the timestamps represent the timing at which the events occurred.

Agents and Messages. Participants in a protocol execution are called *agents*. The set of agents is denoted by Agent, and $\{\mathsf{Honest}, \mathsf{Dishonest}\}$ is a partition of the set of agents into honest and dishonest agents.

During a protocol execution, agents exchange messages through the network. Basic messages are agent names (Agent), nonces (Nonce), and constants (Const). More complex messages can be defined by using atomic messages as the arguments of a function, by pairing them together into a single message or by denoting an encrypted message. Formally, the set of messages Msg is defined by the following grammar, where $atom \in \mathsf{Const} \cup \mathsf{Agent} \cup \mathsf{Nonce}$ and $f \in \mathcal{F}$ are terminal symbols and $\mathcal{F}$ is a countably infinite set of function symbols.

$$\mathsf{Msg} ::= atom \mid \langle \mathsf{Msg}, \mathsf{Msg} \rangle \mid \{\mathsf{Msg}\}_{\mathsf{Msg}} \mid f\,(\mathsf{Msg})\,.$$

The term $\langle m_1, m_2 \rangle$ denotes the pairing of messages m_1 and m_2. Further, $\{m_1\}_{m_2}$ stands for the encryption of m_1 with the key m_2. An agent's signature on a message is represented by the encryption of the message with the secret key of the agent. Finally, $f(m_1)$ indicates the output of the function f on the input m_1. Functions with multiple arguments can be represented through pairing of arguments.

Agents' cryptographic keys are denoted by the functions $pk\colon \mathsf{Agent} \to \mathsf{Msg}$, $sk\colon \mathsf{Agent} \to \mathsf{Msg}$ and $sh\colon \mathsf{Agent} \times \mathsf{Agent} \to \mathsf{Msg}$ that indicate the asymmetric public key of an agent, asymmetric secret key of an agent and the symmetric shared key of two agents, respectively. Lastly, the function $\_^{-1}\colon \mathsf{Msg} \to \mathsf{Msg}$ maps an encryption key onto the corresponding decryption key, and vice-versa.

The set $\mathcal{B} = \{sk, pk, sh, \_^{-1}\} \subseteq \mathcal{F}$ is the set of *basic functions* and its functions are assumed to satisfy that $sh(A, B) = sh(B, A)$, and $pk(A)^{-1} = sk(A)$, and $sk(A)^{-1} = pk(A)$; for all $A, B \in \mathsf{Agent}$. We also assume that, for all $k \in \mathsf{Msg}$, if no $A \in \mathsf{Agent}$ exists such that $k \in \{pk(A), sk(A)\}$, then $k^{-1} = k$. These assumptions represent the properties for symmetric and asymmetric encryption and decryption.

Events and Traces. An event denotes an agent's action, such as sending or receiving a message, or an agent's security claim. We define the set of events Event via the following grammar, for $A, B \in \mathsf{Agent}$.

$$\mathsf{Event} ::= \mathsf{send}\,(A, \mathsf{Msg})\,[\mathsf{Msg}] \mid \mathsf{recv}\,(A, \mathsf{Msg}) \mid \mathsf{claim}\,(A, B, \mathsf{Event}, \mathsf{Event})\,.$$

Given messages m_1 and m_2, and agents A and B, $\mathsf{send}\,(A, m_1)\,[m_2]$ indicates that A has sent the message m_1 and updated the agent's local state with the message m_2, and $\mathsf{recv}\,(A, m_1)$ means that A has received m_1. In the original model, claiming events have the form $\mathsf{claim}\,(A, B, d)$, where $d \in \mathbb{R}$ is a distance value. This allows an agent A to claim

$$\frac{m \in InitK(A)}{m \in DM_A(\alpha)} \qquad \frac{(t, \mathsf{recv}(A, m)) \in \alpha}{m \in DM_A(\alpha)} \qquad \frac{\langle m_1, m_2 \rangle \in DM_A(\alpha)}{\{m_1, m_2\} \subseteq DM_A(\alpha)}$$

$$\frac{\begin{array}{c} m \in DM_A(\alpha) \\ f \in \mathcal{F} \setminus \mathcal{B} \end{array}}{f(m) \in DM_A(\alpha)} \qquad \frac{\begin{array}{c} m_1 \in DM_A(\alpha) \\ m_2 \in DM_A(\alpha) \end{array}}{\langle m_1, m_2 \rangle \in DM_A(\alpha)} \qquad \frac{\begin{array}{c} m \in DM_A(\alpha) \\ k \in DM_A(\alpha) \end{array}}{\{m\}_k \in DM_A(\alpha)} \qquad \frac{\begin{array}{c} \{m\}_k \in DM_A(\alpha) \\ k^{-1} \in DM_A(\alpha) \end{array}}{m \in DM_A(\alpha)}$$

Figure 5.1 Rules for message deduction.

that another agent B is within a radius of length d, which is computed based on the round-trip time of a message exchange. We will make the message exchange explicit, and use $\mathsf{claim}(A, B, e_1, e_2)$ where e_1 and e_2 are the events used to compute the round-trip time and, by extension, the distance bound d.

We define the sets $\mathsf{Send}, \mathsf{Recv} \subseteq \mathsf{Event}$ of all send and receive events, respectively. The function $actor\colon \mathsf{Event} \to \mathsf{Agent}$ maps events onto their corresponding actor agent (i.e., the instance of A from the syntax). We extend this notation by using $actor(\alpha)$, for a given trace α, to refer to the set $\{actor(e) \mid (t, e) \in \alpha\}$. We require for an event $\mathsf{claim}(A, B, e_1, e_2)$ that A is the agent actor of both events e_1 and e_2, i.e. $actor(e_1) = actor(e_2) = A$.

A trace α is a finite sequence of timed-events $\alpha \in (\mathbb{R} \times \mathsf{Event})^*$, representing the execution of a protocol.

Agents' Knowledge. As the trace evolves, agents may gain knowledge by receiving messages from other agents. At the beginning of a protocol execution, every agent is provided with an *initial knowledge* consisting of all agents' names and constants, his own nonces and secret keys, and all public keys. We use the function $InitK\colon \mathsf{Agent} \to \mathcal{P}(\mathsf{Msg})$ to represent the initial knowledge of an agent:

$$\begin{aligned} InitK(A) = \mathsf{Agent} \cup \mathsf{Const} \cup \mathsf{Nonce}_A \cup \{sk(A)\} \cup \\ \{pk(B) \mid B \in \mathsf{Agent}\} \cup \{sh(A, B) \mid B \in \mathsf{Agent}\}, \end{aligned}$$

where Nonce_A denotes the set of nonces for a given agent $A \in \mathsf{Agent}$. We assume that $\{\mathsf{Nonce}_A | A \in \mathsf{Agent}\}$ forms a partition of the set Nonce.

Given an agent A and a trace α, $deducible_A(\alpha)$ denotes the set of all *deducible messages* from a trace α. This set is inductively defined by the rules in Figure 5.1.

Network and Intruder. For a given protocol *Proto*, the set of possible traces $Tr(Proto)$ is inductively defined by the Start rule (`Start`), the Intruder rule (`Int`), the Network rule (`Net`) and the rules specifying the protocol. The Start, Intruder and Network rules are depicted in Figure 5.2.

The rules make use of the function $maxt\colon (\mathbb{R} \times \mathsf{Event})^* \to \mathbb{R}$, which is defined by $maxt(\alpha) = \max_{(t,e)\in\alpha}\{t\}$. That is, $maxt(\alpha)$ yields the latest time at which an event of α occurred. The expression $dist(A, B)$ gives the distance between two agents A and B based on an uninterpreted function $location\colon \mathsf{Agent} \to \mathbb{R}^3$, which associates each agent to a location in the real coordinate space $\mathbb{R}^3$. It is worth remarking that this interpretation of location assumes that agents are static, including dishonest agents.

$$\frac{}{\epsilon \in Tr\,(Proto)}\ \texttt{Start}$$

$$\frac{\begin{array}{c}\alpha \in Tr\,(Proto) \quad t \geq maxt(\alpha) \\ E \in \mathsf{Dishonest} \\ m \in DM_E\,(\alpha)\end{array}}{\alpha \cdot (t, \mathsf{send}\,(E, m)\,[\,]) \in Tr\,(Proto)}\ \texttt{Int} \qquad \frac{\begin{array}{c}\alpha \in Tr\,(Proto) \quad t \geq maxt(\alpha) \\ (t', \mathsf{send}\,(A, m)\,[s]) \in \alpha \\ t \geq t' + dist(A, B)/\mathsf{c}\end{array}}{\alpha \cdot (t, \mathsf{recv}\,(B, m)) \in Tr\,(Proto)}\ \texttt{Net}$$

Figure 5.2 Start, Intruder and Network rules.

The Start rule states that the empty trace ϵ is always part of the set of traces. The Intruder rule enables a dishonest agent, typically known as the *intruder* or the *adversary*, to inject (by sending) on the network any of his deducible messages. Finally, the Network rule establishes that a message m sent by an agent A can be received by an agent B without violating a time/location constraint that we describe in the next paragraph. This constraint is actually what makes this model particularly different from standard security models.

The Network rule also states that a message sent by an agent A and received by an agent B at times t' and t, respectively, must satisfy $dist(A, B) \leq (t - t') \cdot \mathsf{c}$. In this way the physical law that messages cannot travel faster than the speed of light is made explicit. Observe that message loss is captured by *not* applying the network rule for a given sending event.

Protocol Specification. A protocol is specified by a set of rules similar to the rules in Figure 5.2. Two syntactic restrictions[1] are applied:

(1) Neither the premises nor the conclusion of a protocol rule contain references to dishonest agents. This means that the behavior of dishonest agents is fully specified by the intruder rule.

(2) The premise of a protocol rule cannot contain events whose actors are not the same as the actor of the event in the premise of the rule. That is to say, agents are unaware of what other agents do. They can interact exclusively through the network rule.

Example 5.1 (Hancke and Kuhn's protocol). Recall the HK protocol from Figure 1.5 whose symbolic representation and specification rules are shown in Figures 5.3 and 5.4, respectively. The first four rules in Figure 5.4 correspond to the four transmissions that take place in the protocol. The receive events are derived from the network rule. The last rule from Figure 5.4 refers to the claim event for the property *secure distance-bounding* represented as "P is close" in Figures 5.3. The function $subterm\colon \mathsf{Msg} \to \mathcal{P}\,(\mathsf{Msg})$ yields the set of atomic messages that are sub-terms of a given message, and it is recursively defined by:

$$subt(m) = \begin{cases} subt(m_1) \cup subt(m_2) & \text{if } m = \langle m_1, m_2 \rangle \text{ or } m = \{m_1\}_{m_2} \\ subt(m_1) & \text{if } m = f(m_1) \\ \{m\} & \text{otherwise .} \end{cases}$$

[1] Semantic interpretations of these restrictions will be given in Section 5.2.1.

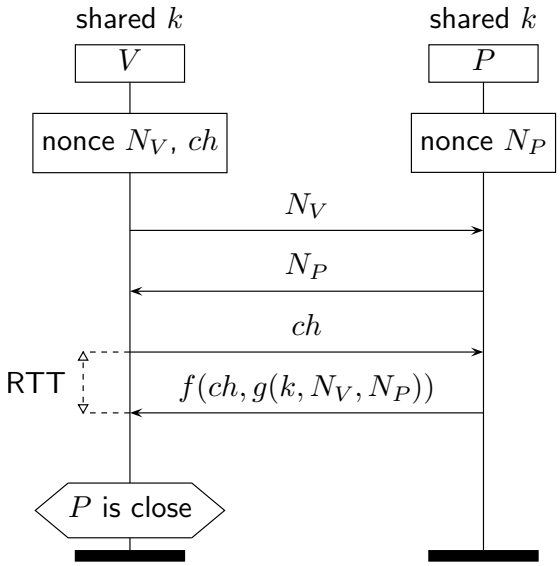


Figure 5.3 Symbolic abstraction of Hancke and Kuhn's protocol [HK05]. Differences in notation with respect to the representation in Figure 1.5 are two fold: ch instead of c to avoid confusion with the transmission speed symbol c, and g instead of PRF for the sake of presentation.

Furthermore, the function $used\colon (\mathbb{R} \times \mathsf{Event})^* \to \mathcal{P}(\mathsf{Msg})$ is utilized to make sure that newly generated nonces are fresh, and it is defined by:

$$used(\alpha) = \bigcup_{(t,e)\in\alpha} subt(content\,(e)),$$

where $content\colon \mathsf{Event} \to \mathsf{Msg}$ gives us the content of a given event.

Example 5.1 also illustrates the purpose of the information in square brackets at the end of the send actions. In this case, it is implicitly used to define the notion of a session, by extending the send actions with the random nonces from that session. Further, it is used to specify in which order the events of a session will have to be executed.

Security Properties. The model uses claim events as placeholders to indicate where a security property needs to be satisfied. In this paper we focus on the property of *secure distance-bounding*, which is syntactically represented by claims of the form $\mathsf{claim}\,(V, P, x, y)$, where $V, P \in \mathsf{Agent}$ and $x, y \in \mathsf{Event}$. A claim event $\mathsf{claim}\,(V, P, x, y)$ intuitively means that the agent V believes that the events x and y can be used to correctly compute an upper bound on his distance to P.

As the Intruder rule suggests, dishonest agents might disclose their secret key material by sending them out. This means that two dishonest provers might be indistinguishable to a legitimate verifier. In other words, a verifier V cannot securely decide whether a particular dishonest prover P is close, as another dishonest prover P' could have obtained all P's secrets and therefore P' can impersonate P. This leads to the following statement: V cannot claim that "P is close" but V can claim that "someone who knows P's secrets is close", at most. To capture this notion, we define the relation $\approx = \{(A, A) \mid A \in \mathsf{Honest}\} \cup \mathsf{Dishonest} \times \mathsf{Dishonest}$.

$$\frac{\alpha \in Tr\,(Proto) \quad V \in \mathsf{Honest} \quad t \geq maxt(\alpha) \quad N_V \in \mathsf{Nonce}_V \setminus used(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(V, N_V)\,[\,]) \in Tr\,(Proto)}$$

$$\frac{\alpha \in Tr\,(Proto) \quad P \in \mathsf{Honest} \quad t \geq maxt(\alpha) \quad (t', \mathsf{recv}\,(P, N_V)) \in \alpha \quad N_P \in \mathsf{Nonce}_P \setminus used(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(P, N_P)\,[N_V]) \in Tr\,(Proto)}$$

$$\frac{\alpha \in Tr\,(Proto) \quad V \in \mathsf{Honest} \quad t \geq maxt(\alpha) \quad (t', \mathsf{send}\,(V, N_V)\,[\,]) \in \alpha \quad (t'', \mathsf{recv}\,(V, N_P)) \in \alpha \quad ch \in \mathsf{Nonce}_V \setminus used(\alpha)}{\alpha \cdot (t, \mathsf{send}\,(V, ch)\,[N_V, N_P]) \in Tr\,(Proto)}$$

$$\frac{\alpha \in Tr\,(Proto) \quad P \in \mathsf{Honest} \quad t \geq maxt(\alpha) \quad (t', \mathsf{send}\,(P, N_P)\,[N_V]) \in \alpha \quad (t'', \mathsf{recv}\,(P, ch)) \in \alpha \quad rp = f(ch, g(sh(V, P), N_V, N_P))}{\alpha \cdot (t, \mathsf{send}\,(P, rp)\,[\,]) \in Tr\,(Proto)}$$

$$\frac{\alpha \in Tr\,(Proto) \quad V \in \mathsf{Honest} \quad tz \geq maxt(\alpha) \quad rp = f(ch, g(sh(V, P), N_V, N_P)) \quad x = \mathsf{send}\,(V, ch)\,[N_V, N_P] \quad y = \mathsf{recv}\,(V, rp) \quad (tx, x) \in \alpha \quad (ty, y) \in \alpha}{\alpha \cdot (tz, \mathsf{claim}\,(V, P, x, y)) \in Tr\,(Proto)}$$

Figure 5.4 Specification rules of Hancke and Kuhn's protocol [HK05].

Definition 5.1 (Secure Distance-Bounding)**.** A protocol *Proto* satisfies *secure distance-bounding* if and only if:

$$\begin{aligned} &\forall \alpha \in Tr\,(Proto)\,, (tz, \mathsf{claim}\,(V, P, x, y)) \in \alpha. \\ &\quad \exists (tx, x), (ty, y) \in \alpha, P' \in actor\,(\alpha)\,. \\ &\qquad P \approx P' \wedge\ dist(V, P') \leq \mathsf{c} \cdot (ty - tx)\,/2. \end{aligned} \tag{5.1}$$

A distance-bounding protocol is secure if the occurrence of a claim event $\mathsf{claim}\,(V, P, x, y)$ in a protocol execution implies that V has correctly computed an upper bound on his distance to either P (if P is honest) or some dishonest agent P' (if P is dishonest).

The definition of secure distance-bounding slightly differs from the original one provided by Basin et al., but the difference is merely notational, allowing us to cleanly formulate our main result in Section 5.3. Note that claim events are formulated in a way that they relate to a single challenge/response pair. Thus, similar to Basin et al.'s approach, we will need to include several claim events if the fast phase cannot be abstracted to a single challenge/response pair.

5.2 The Semantic Domain

An important characteristic of Basin et al.'s approach, as presented in the previous section, is that security protocols are specified using the same type of derivation rules as used for the definition of the general semantics of the system. Consequently, protocol specifications are much more liberal than in comparable formal approaches that define a domain specific language for the definition of protocols. Alternative approaches, like the one by Cremers and Mauw [CM12] provide a dedicated protocol specification language and impose syntactical

$$\frac{\begin{array}{c}\alpha \in Tr\,(Proto) \qquad A \in \mathsf{Honest} \\ \mathsf{Hello}, \mathsf{Hi} \in \mathsf{Const} \\ (t, \mathsf{recv}\,(A, \mathsf{Hello})) \in \alpha\end{array}}{\alpha \cdot (t-1, \mathsf{send}\,(A, \mathsf{Hi})\,[\,]) \in Tr\,(Proto)}$$

Figure 5.5 A protocol rule that leads to incorrect traces.

or semantical constraints to prevent users from specifying meaningless or simply undesired protocols.

An example of a protocol rule that may be considered undesirable is the one in Figure 5.5. It specifies that after reception of the message Hello at time t, agent A sends a message Hi back at time $t-1$. This is clearly an infringement of a time consistency property, because it leads to the trace $(1, \mathsf{recv}\,(A, \mathsf{Hello})) \cdot (0, \mathsf{send}\,(A, \mathsf{Hi})\,[])$.

The solution proposed by Basin et al. is to consider only those traces that have non-decreasing timestamps for subsequent events. In our approach we will take this line of reasoning one step further, in that we will define a number of assumptions that a proper semantics should satisfy and that are sufficient to derive our main result. We will argue that these properties are valid for the semantics from the previous section, under the assumption of a class of "reasonable" protocol specifications.

5.2.1 Basic Properties of the Semantics

In line with the previous example, the first property that we formulate is *time consistency*. It states that events of a trace are timestamped in non-decreasing order.

Property 5.1 (Time consistency). A protocol *Proto* satisfies *time consistency* if $t_1 \leq \ldots \leq t_n$ for all traces $(t_1, e_1) \cdots (t_n, e_n) \in Tr\,(Proto)$.

The second property that we consider is *speed-of-light consistency*. It states that all traces satisfy the restrictions of the speed of light. In particular, this means that the time between the sending of a message by agent A and the reception of this message by agent B must be equal to or larger than the distance between the two agents divided by the speed of light. Because this definition requires the correspondence between a send event and its related receive event, we define the relation $\rightsquigarrow \subseteq \mathsf{Send} \times \mathsf{Recv}$ as follows:

$$\rightsquigarrow = \left\{(e, e') \in \mathsf{Send} \times \mathsf{Recv} \mid content\,(e) = content\,\left(e'\right)\right\}.$$

The relation $\rightsquigarrow$ defines whether an event e' is a receive event that could have occurred as consequence of the send event e. As followed from its formulation, $\rightsquigarrow$ is not a one-to-one relation. This lines up with the fact that it does not need to be the case that there is a unique send event that triggers a given receive event. In the semantics above, the relation $\rightsquigarrow$ can be easily derived from the application of the Network rule in Figure 5.2.

Property 5.2 (Speed-of-light consistency). A protocol *Proto* satisfies *speed-of-light consistency* if for every trace $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in Tr\,(Proto)$ and for all $j \in \{2, \ldots, n\}$ such that $e_j \in \mathsf{Recv}$, there exists $i \in \{1, \ldots, j-1\}$ such that $e_i \rightsquigarrow e_j$ and $t_j - t_i \geq \dfrac{dist(e_i, e_j)}{\mathsf{c}}$.

Even though we define Properties 5.1 and 5.2 for protocols, we will also use them in relation to traces. Thus we will talk about time consistency and speed-of-light consistency of a given trace, with the obvious interpretation.

The formulation of the remaining properties requires the notion of *untimed* traces, or simply a sequence of (untimed) events. The projection of a trace $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in (\mathbb{R} \times \mathsf{Event})^*$ is the untimed untimed trace $e_1 \cdots e_n \in \mathsf{Event}^*$. Likewise, the projection of the set of traces is defined as $\pi(\mathit{Tr}\,(\mathit{Proto})) = \{\pi(\alpha) \mid \alpha \in \mathit{Tr}\,(\mathit{Proto})\}$. We say that two traces α and β are content-wise equal, denoted $\alpha \approx \alpha'$, if $\pi(\alpha) = \pi(\beta)$.

The third property states that traces are built inductively by appending events.

Property 5.3 (Prefix-closure)**.** A protocol *Proto* is *prefix-closed* if for every untimed trace $\gamma = \sigma \cdot e \in \pi(\mathit{Tr}\,(\mathit{Proto}))$, it holds that $\sigma \in \pi(\mathit{Tr}\,(\mathit{Proto}))$.

The fourth property expresses that the notion of time is only used for the verifier's decision-making process on whether the prover passed the protocol or not. Time will not be used to make any other decision during the execution of the protocol (e.g., to take a different branch depending on the time). This means that any trace can be *retimed*, as long as it still satisfies time consistency and speed-of-light consistency.

Property 5.4 (Time-unawareness)**.** A protocol *Proto* is *time-unaware* if for every $\alpha \in \mathit{Tr}\,(\mathit{Proto})$ and for all time consistent and speed-of-light consistent traces $\beta \in (\mathbb{R} \times \mathsf{Event})^*$ such that $\alpha \approx \beta$ it holds that $\beta \in \mathit{Tr}\,(\mathit{Proto})$.

As mentioned in Section 5.1, different agents only interact through the network via sending and receiving events. As a consequence, a non-receive action can only be triggered by the actor agent's own preceding actions and another agent's actions in between can be disregarded or delayed. This leads to the fifth property, *locally-enabled events*. We use untimed events in order to easily express that the resulting trace $\sigma \cdot e'$ might require a re-timing of event e'.

Property 5.5 (Locally-enabled events)**.** A protocol *Proto* satisfies *locally-enabled events* if for every $\gamma = \sigma \cdot e \cdot e' \in \pi(\mathit{Tr}\,(\mathit{Proto}))$ such that $e' \notin \mathsf{Recv}$ and $\mathit{actor}\,(e) \neq \mathit{actor}\,(e')$, it holds that $\sigma \cdot e' \in \pi(\mathit{Tr}\,(\mathit{Proto}))$.

The *locally-enabled events* property allows non-receive events to move left in a trace under specific conditions. The next property expresses when a receive event can be appended to a trace.

Property 5.6 (Transmission-enabled events)**.** A protocol *Proto* satisfies *transmission-enabled events* if for every $\gamma = \sigma \cdot e \in \pi(\mathit{Tr}\,(\mathit{Proto}))$ and every $e' \in \mathsf{Recv}$ such that $e \rightsquigarrow e'$, it holds that $\gamma \cdot e' \in \pi(\mathit{Tr}\,(\mathit{Proto}))$.

Agents in the model are universally quantified. Therefore, in a given trace we can replace an agent by another and still obtain a valid trace, as long as both agents are either honest or dishonest. An agent substitution is denoted by $A \mapsto B$ where A and B are agents. Given a message $m \in \mathsf{Msg}$, $m[A \mapsto B]$ represents the substitution of all occurrences in m of A by B. We extend substitutions onto events and traces in the obvious way.

$$\frac{\alpha \in Tr\,(Proto) \quad A \in \mathsf{Honest} \quad t \geq maxt(\alpha) \quad prem_1 \quad prem_2 \quad \cdots \quad prem_p \quad (t_1, e_1) \in \alpha \quad (t_2, e_2) \in \alpha \quad \cdots \quad (t_q, e_q) \in \alpha}{\alpha \cdot (t, e) \in Tr\,(Proto)}$$

Figure 5.6 Prototype of rules that lead to well-formed protocols.

Property 5.7 (Substitution-closure). A protocol *Proto* is *substitution-closed* if for every untimed trace $\sigma \in \pi(Tr\,(Proto))$ and every $A, B \in \mathsf{Agent}$ such that $\{A, B\} \subseteq \mathsf{Honest}$ or $\{A, B\} \subseteq \mathsf{Dishonest}$, it holds that $\sigma[A \mapsto B] \in \pi(Tr\,(Proto))$.

Observe that $e \rightsquigarrow e'$ implies $e[A \mapsto B] \rightsquigarrow e'[A \mapsto B]$. We say that a protocol is *well-formed* if it satisfies the seven properties mentioned above.

5.2.2 Validity of the Properties

As stated before, the mechanism for specifying protocols is too liberal to ensure the well-formedness properties. Therefore, we use a restricted format for protocol rules inspired by the example specification of Hancke and Kuhn's protocol from Figure 5.4. The restricted format is specified by the rule prototype in Figure 5.6. We additionally require that $p + q > 0$, $A = actor\,(e) = actor\,(e_1) = actor\,(e_2) = \cdots = actor\,(e_q)$, $e \notin \mathsf{Recv}$ and none of the premises $prem_i$ involve any of the timestamps t_j or t. Even though the protocol format is restricted with respect to the liberal format specified by Basin et al., we conjecture that it is sufficiently expressive to specify all relevant protocols from literature. We validate this by specifying a number of protocols in this format and analysing them with our implementation (see Section 5.3).

Together with the Start, Intruder and Network rules from Figure 5.2, the restricted format implies well-formedness of the specified protocol. We briefly argue next the validity of the properties under this restricted format.

Time consistency follows from the precondition $t \geq maxt(\alpha)$ in the Intruder and Network rules and in the restricted protocol rule. Speed-of-light consistency follows from the precondition $t \geq t' + dist(A, B)/\mathsf{c}$ in the Network rule and the requirement that $e \notin \mathsf{Recv}$ in the restricted protocol rule. Prefix-closure follows from the precondition $\alpha \in Tr\,(Proto)$ in all rules, together with the fact that the conclusion extends this trace with a single event. Time-unawareness follows from the fact that in the construction of the traces any time $t \geq maxt(\alpha)$ is allowed for the next event, as long as *speed-of-light consistency* is satisfied. The property locally-enabled events follows from the requirement that a rule only concerns a single actor. The transmission-enabled events property follows directly from the Network rule. Substitution-closure expresses the (implicit) universal quantification over agents' names in all rules.

5.3 Causality-Based Verification

Given the definitions and properties from the previous sections, we can now formulate the notion of *causality-based secure distance-bounding* and prove that it is equivalent to

the original definition of *secure distance-bounding* from Definition 5.1. The main feature of this new formulation is that it is causality-based, i.e., it only takes into account the relative occurrence of events, while ignoring the actual timestamps of the events and agents' locations.

This new formulation resembles authentication properties, such as *aliveness* [Low97, CM12]. It states that for every claim that prover P is in the vicinity of verifier V, due to a challenge event x and the reception of its corresponding response event y in the fast phase, agent P (or a conspiring agent, if P is dishonest) must have been active in between these two events. The main difference with Definition 5.1 is that we require the prover to be active, instead of measuring the time between x and y.

Definition 5.2 (Causality-based Secure Distance-Bounding)**.** A well-formed protocol *Proto* satisfies *causality-based secure distance-bounding* if and only if:

$$\begin{aligned}&\forall \sigma \in \pi(\mathit{Tr}\,(\mathit{Proto})), \mathsf{claim}\,(V, P, x, y) \in \sigma.\\ &\quad \exists i, j, k \in \{1, \ldots, |\sigma|\}.\\ &\qquad i < j < k \wedge\ \sigma_i = x \wedge\ \sigma_k = y \wedge\ P \approx \mathit{actor}\,(\sigma_j)\,.\end{aligned} \tag{5.2}$$

In Definition 5.2 we formalize our causality-based notion of secure distance-bounding. This formulation impacts only the security analysis in the design stage. It does not affect the runtime behavior of the agents executing the protocol. In particular, the verifying agent still has to measure the round-trip time of the message exchanges in the fast phase.

In the remainder of this section, we develop the proof that the causality-based definition is equivalent to the secure distance-bounding property from Definition 5.1. To do so, we first present a few lemmas that follow from the basic properties of the semantic domain described in Section 5.2.1. They will prove useful when deriving our main result.

Given two events $e, e' \in \mathsf{Event}$, we will write $\mathit{dist}(e, e')$ as a shorthand notation for $\mathit{dist}(\mathit{actor}\,(e)\,, \mathit{actor}\,(e'))$. Also, we say that two timed-events $(t, e), (t', e') \in \mathbb{R} \times \mathsf{Event}$ satisfy the time/location constraint if $|t' - t| \geq \dfrac{\mathit{dist}(e, e')}{\mathsf{c}}$. For example, all pairs of events used in the network rule satisfy this constraint. In addition, we define the predicate $\psi\,(\alpha)$, where α is a trace, that holds if all pairs of consecutive timed-events on α satisfy the time/location constraint. Likewise, we say that timed-trace β is a subsequence of a timed-trace $\alpha = (t_1, e_1) \cdots (t_n, e_n)$, denoted by $\beta \sqsubseteq \alpha$, if there exist $m \in \{0, \ldots, n\}$ and $\{w_1, \ldots, w_m\} \subseteq \{1, \ldots, n\}$ such that $w_1 < \cdots < w_m$ and $\beta = (t_{w_1}, e_{w_1}) \cdots (t_{w_m}, e_{w_m})$.

In Lemma 5.1 below, we demonstrate that for any well-formed protocol *Proto*, any valid timed-trace $\alpha \cdot (t, e) \in \mathit{Tr}\,(\mathit{Proto})$ must contain a subsequence β that is also a valid trace in *Proto*, and contains (t, e) and $\psi\,(\beta)$. We use $|.|$ to denote the length of a (timed or not) trace, in terms of the number of events.

Lemma 5.1. Let *Proto* be a well-formed protocol, then the following holds:

$$\begin{aligned}&\forall \alpha \in \mathit{Tr}\,(\mathit{Proto})\,, (t, e) \in \mathbb{R} \times \mathsf{Event}.\\ &\quad \alpha \cdot (t, e) \in \mathit{Tr}\,(\mathit{Proto}) \implies\\ &\qquad \exists \beta \in \mathit{Tr}\,(\mathit{Proto})\,.\ (t, e) \in \beta\ \wedge \beta \sqsubseteq \alpha \cdot (t, e)\ \wedge \psi\,(\beta)\,.\end{aligned}$$

Proof. We will proceed by induction over $|\alpha|$. The base case $|\alpha| = 0$ trivially holds by setting $\beta = (t, e)$. So, let $n \in \mathbb{Z}_{>0}$ and assume by the induction hypothesis that the lemma

holds for all $\alpha \in Tr\,(Proto)$ with $|\alpha| < n$. Now, let $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in Tr\,(Proto)$ and $(t, e) \in \mathbb{R} \times \mathsf{Event}$ such that $\gamma = \alpha \cdot (t, e) \in Tr\,(Proto)$. Let us analyze the two cases:

Case $e \in \mathsf{Recv}$: From Property 5.2 we have that there exists $i \in \{1, \ldots, n\}$ such that $e_i \rightsquigarrow e$ and $t - t_i \geq \dfrac{dist(e_i, e)}{\mathsf{c}}$. Consider $\alpha' = (t_1, e_1) \cdots (t_{i-1}, e_{i-1})$. Then, from the induction hypothesis (given that $|\alpha'| = i - 1 < n$ and $\alpha' \in Tr\,(Proto)$ due to Properties 5.3 and 5.4) it follows that there exists $\beta' \in Tr\,(Proto)$ with $(t_i, e_i) \in \beta'$ such that $\beta' \sqsubseteq \alpha'$ and $\psi\,(\beta')$. Thus, $\psi\,(\beta')$ along with $t - t_i \geq \dfrac{dist(e_i, e)}{\mathsf{c}}$ give us $\psi\,(\beta' \cdot (t, e))$ and that $\beta' \cdot (t, e)$ is time and speed-of-light consistent.

Hence, from Property 5.6 we derive $\pi(\beta') \cdot e \in \pi(Tr\,(Proto))$. Further, $\beta' \cdot (t, e) \approx \beta''$ for some $\beta'' \in Tr\,(Proto)$ such that $\pi(\beta') \cdot e = \pi(\beta'')$. Finally, Property 5.4 gives us $\beta' \cdot (t, e) \in Tr\,(Proto)$.

Case $e \notin \mathsf{Recv}$: Let i be the largest number in $\{1, \ldots, n\}$ such that $actor\,(e_i) = actor\,(e)$. If i does not exist, then from Property 5.5 we obtain that $e \in \pi(Tr\,(Proto))$ and therefore $(t', e) \in Tr\,(Proto)$ for some $t' \in \mathbb{R}$. Hence, as (t, e) is time and speed-of-light consistent, Property 5.4 gives us $(t, e) \in Tr\,(Proto)$ as $(t, e) \approx (t', e)$. Further, $\psi\,((t, e))$ trivially holds, which leaves us with the remaining case in which i exists.

Let $\alpha' = (t_1, e_1) \cdots (t_i, e_i)$. Then, from the induction hypothesis (given that $|\alpha'| = i - 1 < n$ and $\alpha' \in Tr\,(Proto)$ due to Properties 5.3 and 5.4) it follows that there exists $\beta' \in Tr\,(Proto)$ with $(t_i, e_i) \in \beta'$ such that $\beta' \sqsubseteq \alpha'$ and $\psi\,(\beta')$. Thus, $\psi\,(\beta')$ along with $t - t_i \geq \dfrac{dist(e_i, e)}{\mathsf{c}} = 0$ give us $\psi\,(\beta' \cdot (t, e))$ and that $\beta' \cdot (t, e)$ is time and speed-of-light consistent.

Hence, from Property 5.5[2] we derive $\pi(\beta') \cdot e \in \pi(Tr\,(Proto))$. Further, $\beta' \cdot (t, e) \approx \beta''$ for some $\beta'' \in Tr\,(Proto)$ such that $\pi(\beta'') = \pi(\beta') \cdot e$. Thus, Property 5.4 gives us $\beta' \cdot (t, e) \in Tr\,(Proto)$.

□

Lemma 5.2 below is an extension of Lemma 5.1. It states that if a valid trace α satisfies $\psi\,(\alpha)$, then not only any pair of consecutive events in α satisfy the time/location constraint but also any pair of events in α. The proof follows from the application of the triangle inequality $dist(e, e') + dist(e', e'') \geq dist(e, e'')$ for all $e, e', e'' \in \mathsf{Event}$, given that $dist$ models physical distances.

Lemma 5.2. Let $Proto$ be a well-formed protocol and $\alpha \in Tr\,(Proto)$ such that $\psi\,(\alpha)$. Then $|t - t'| \geq \dfrac{dist(e, e')}{\mathsf{c}}$ for all $(t, e), (t', e') \in \alpha$.

Proof. Let $\alpha = (t_1, e_1) \cdots (t_n, e_n)$ and $i, j \in \{1, \ldots, n\}$. Assume without loss of generality that $i < j$. From $\psi\,(\alpha)$ it follows that, for all $w \in \{i, \ldots, j - 1\}$:

$$t_w - t_{w-1} \geq \frac{dist(e_w, e_{w+1})}{\mathsf{c}}$$

[2] In published version, Property 5.6 is referenced here instead, which is a typo.

Hence,

$$t_j - t_i = (t_j - t_{j-1}) + (t_{j-1} - t_{j-2}) + \cdots + (t_{i+1} - t_i) \geq \sum_{w=i}^{j-1} \frac{dist(e_w, e_{w+1})}{\mathsf{c}}. \tag{5.3}$$

Hence, by applying the triangle inequality above, we obtain $t_j - t_i \geq \frac{dist(e_i, e_j)}{\mathsf{c}}$. □

The last lemma of this section refers to agent substitutions. We extend Property 5.7 from the set of untimed-traces $\pi(Tr\,(Proto))$ of a given protocol $Proto$ to the set of timed-traces $Tr\,(Proto)$. The lemma proves that, given a protocol's valid trace $\alpha = (t_1, e_1) \cdots (t_n, e_n)$, it is possible to replace an agent A by another agent B (under certain conditions described in the lemma) to obtain another valid trace $\alpha' = (t'_1, e'_1) \cdots (t'_n, e'_n)$ such that the difference between t'_i and t_i only depends on the number of events before the i-th event on α that were executed by A. Consequently, the time-difference between two events of α where A does not act is equal to the time-difference between the corresponding events of α'. This is actually a strong result because it implicitly shows that event-intervals where the prover does not act cannot be used to securely upper-bound the prover-to-verifier distance.

Lemma 5.3. Let $Proto$ be a well-formed protocol and $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in Tr\,(Proto)$. Let $A \in actor\,(\alpha)$ and $B \in \mathsf{Agent} \setminus actor\,(\alpha)$ such that either $\{A, B\} \subseteq \mathsf{Honest}$ or $\{A, B\} \subseteq \mathsf{Dishonest}$. Therefore $\mu \in \mathbb{R}_{\geq 0}$ and $\alpha' = (t'_1, e'_1) \cdots (t'_n, e'_n) \in Tr\,(Proto)$ exist such that, for all $i \in \{1, \ldots, n\}$:

$$\begin{aligned}
&e'_i = e_i[A \mapsto B] \quad \text{and} \quad t'_i = t_i + \mu \cdot q_i, \quad \text{with} \\
&q_i = |\{j \in \{1, \ldots, i-1\} \mid actor\,(e_j) = A\}| + s_i, \quad \text{and} \\
&s_i = \begin{cases} 1 & \text{if } A = actor\,(e_i) \wedge e_i \in \mathsf{Recv} \\ 0 & \text{otherwise .} \end{cases}
\end{aligned}$$

Proof. Consider the set $R = \{B\} \cup actor\,(\alpha)$ and $\mu = \max_{X \in R} \left\{ \frac{dist(A, X)}{\mathsf{c}} \right\}$. We will proceed to prove that $\alpha' \in Tr\,(Proto)$. To do so we will first prove time consistency and speed-of-light consistency for α'.

Time consistency: For all $i \in \{1, \ldots, n-1\}$, we have that $q_{i+1} \geq q_i$ and therefore $t'_{i+1} - t'_i = t_{i+1} - t_i + \mu \cdot (q_{i+1} - q_i) \geq t_{i+1} - t_i \geq 0$.

Speed-of-light consistency: Let $j \in \{1, \ldots, n\}$ such that $e_j \in \mathsf{Recv}$. Also, as α is speed-of-light consistent, we derive that there exists $i < j$ such that $e_i \rightsquigarrow e_j$ and $t_j - t_i \geq \frac{dist(e_i, e_j)}{\mathsf{c}}$. Hence, given that $e'_i \rightsquigarrow e'_j$, it becomes sufficient to prove that $t'_j - t'_i \geq \frac{dist(e'_i, e'_j)}{\mathsf{c}}$. Let us consider the three cases:

Case $A = actor\,(e_i)$: In this case $q_j \geq q_i + 1$ because $e_i \notin \mathsf{Recv}$. Therefore $t'_j - t'_i \geq t_j - t_i + \mu \geq \frac{dist(e'_i, e'_j)}{\mathsf{c}}$ as $\mu \geq \frac{dist(e'_i, e'_j)}{\mathsf{c}}$.

Case $A \neq actor\,(e_i)$ and $A = actor\,(e_j)$: In this case we have again $q_j \geq q_i + 1$ as $e_j \in \mathsf{Recv}$, and it follows analogously to the previous case.

Case $A \notin \{actor\,(e_i)\,, actor\,(e_j)\}$: This case gives us $actor\,(e_i) = actor\,(e'_i)$ and $actor\,(e_j) = actor\,(e_j)$. Thus, $\frac{dist(e_i,e_j)}{\mathsf{c}} = \frac{dist(e'_i,e'_j)}{\mathsf{c}}$ and therefore $t'_j - t'_i = t_j - t_i + \mu \cdot (q_j - q_i) \geq t_j - t_i \geq \frac{dist(e_i,e_j)}{\mathsf{c}} = \frac{dist(e'_i,e'_j)}{\mathsf{c}}$.

Thus, α' is time consistent and speed-of-light consistent. Consider now $\sigma = \pi(\alpha)$. From Property 5.7 we have that $\sigma[A \mapsto B] \in \pi(Tr\,(Proto))$. Therefore, there exists $\gamma \in Tr\,(Proto)$ such that $\pi(\gamma) = \sigma[A \mapsto B]$. Finally, given that $\gamma \approx \alpha'$, from Property 5.4 we obtain $\alpha' \in Tr\,(Proto)$. □

Theorem 5.1. A well-formed protocol *Proto* satisfies *secure distance-bounding* (Definition 5.1) if and only if *Proto* satisfies *causality-based secure distance-bounding* (Definition 5.2).

Proof. We will proceed by proving sufficiency (i.e. Equation 5.1 ⇒ Equation 5.2) and necessity(i.e., Equation 5.2 ⇒ Equation 5.1).

Sufficiency: Assume Equation 5.1 holds and Equation 5.2 does not. Our goal is to reach a contradiction. The statement that Equation 5.2 does not hold is equivalent to stating that there exist $\sigma = \sigma_1 \cdots \sigma_n \in \pi(Tr\,(Proto))$, and $V, P \in \mathsf{Agent}$, and $x, y \in \mathsf{Event}$, and $l \in \{1, \ldots, n\}$ such that $\sigma_l = \mathsf{claim}\,(V, P, x, y)$ and:

$$\begin{aligned} &\forall i, j, k \in \{1, \ldots, n\}. \\ &\quad (x = \sigma_i \wedge y = \sigma_k \wedge i < j < k) \implies P \not\approx actor\,(\sigma_j)\,. \end{aligned} \tag{5.4}$$

Consider now the following sets:

$$\begin{aligned} I &= \{(i,k) \in \mathbb{N} \times \mathbb{N} \mid \sigma_i = x \wedge \sigma_k = y\}, \\ J &= \{j \in \mathbb{N} \mid \exists (i,k) \in I.\ i < j < k\}, \\ \mathcal{G} &= \{G_1, \ldots, G_g\} = \{G \in actor\,(\sigma) \mid P \approx G\}. \end{aligned}$$

If P is honest, then $\mathcal{G} = \{P\}$, otherwise $\mathcal{G}$ contains all dishonest agents that act in σ. Let $Eve, Charlie \in \mathsf{Agent} \setminus actor\,(\sigma)$ be two different agents such that P, Eve and $Charlie$ are either all honest or all dishonest.

Consider the sequence of traces $\sigma^1, \ldots, \sigma^{g+1} \in \pi(Tr\,(Proto))$ such that $\sigma^1 = \sigma$ and $\sigma^{i+1} = \sigma^i[G_i \mapsto Eve]$ for all $i \in \{1, \ldots, g\}$. Observe that $\sigma^1, \ldots, \sigma^{g+1} \in \pi(Tr\,(Proto))$ follows from the *substitution-closedness* property. Hence, let $e_1 \cdots e_n = \sigma^{g+1}$, i.e., the trace resulting from σ after the successive substitutions of all agents $G_1, \ldots, G_g$ by Eve. Therefore $N \subseteq \mathsf{Agent}$ exists such that:

$$actor\,(e_1 \cdots e_n) = \{V, Eve\} \cup N \ \wedge\ \forall E \in N.\ Eve \not\approx E. \tag{5.5}$$

Let $t_1, \ldots, t_n \in \mathbb{R}$ such that $(t_1, e_1) \cdots (t_n, e_n) \in Tr\,(Proto)$. Observe that the t_i's exist because $e_1 \cdots e_n \in \pi(Tr\,(Proto))$. Hence, from Equations 5.1 and 5.5 and given that $e_l = \mathsf{claim}\,(V, Eve, e_i, e_k)$ for some $(i,k) \in I$, we derive that $\delta \in \mathbb{R}_{\geq 0}$ exists such that:

$$dist(V, Eve) + \delta = \frac{\mathsf{c}}{2} \max_{(i,k) \in I} \{t_k - t_i\}. \tag{5.6}$$

From Lemma 5.3 we have that there exist $\mu \in \mathbb{R}_{\geq 0}$, $(t'_1, e'_1) \cdots (t'_n, e'_n) \in Tr\,(Proto)$ and $q_1, \ldots, q_n \in \mathbb{N}$ such that $e'_i = e_i[Eve \mapsto Charlie]$ and $t'_i = t_i + \mu \cdot q_i$ (see the construction of the q_i's in Lemma 5.3) for all $i \in \{1, \ldots, n\}$. On the other hand, from Equation 5.4 we have that $Eve \neq actor\,(e_j)$ for all $j \in J$. Therefore

$$\forall (i,k) \in I.\ t'_k - t'_i = t_k - t_i. \tag{5.7}$$

Furthermore, given that $\{Eve, Charlie\} \subseteq \mathsf{Honest}$ or $\{Eve, Charlie\} \subseteq \mathsf{Dishonest}$, it holds that:

$$actor\,(e'_1 \cdots e'_n) = \{V, Charlie\} \cup N \ \wedge\ \forall C \in N.\ Charlie \not\approx C. \tag{5.8}$$

Again, $e'_l = \mathsf{claim}\,(V, Charlie, e'_i, e'_k)$ for some $(i,k) \in I$, so from Equations 5.1 and 5.8 we derive:

$$dist(V, Charlie) \leq \frac{\mathsf{c}}{2} \max_{(i,k)\in I} \{t'_k - t'_i\}. \tag{5.9}$$

Finally, from Equations 5.6, 5.7 and 5.9 we derive that $dist(V, Charlie) \leq d(V, Eve) + \delta$. This is a contradiction, as δ does not depend on $Charlie$ who is an arbitrary agent (from the same set, either Honest or $\mathsf{Dishonest}$, as P). Therefore we can always find $Charlie$ such that his distance to V is larger than $dist(V, Eve) + \delta$.

Necessity: Assume Equation 5.2 holds. We will prove that Equation 5.1 holds as well. Let $\sigma \in \pi(Tr\,(Proto))$ and $\alpha \in Tr\,(Proto)$ such that $\sigma = \pi(\alpha)$. Let $V, P \in \mathsf{Agent}$, and $x, y \in \mathsf{Event}$ and $tz \in \mathbb{R}$ such that $(tz, \mathsf{claim}\,(V, P, x, y)) \in \alpha$. Furthermore, let $\beta \in Tr\,(Proto)$ such that $\beta \sqsubseteq \alpha$, $(tz, \mathsf{claim}\,(V, P, x, y)) \in \beta$ and $\psi\,(\beta)$. Observe that β exists because of Lemma 5.1.

From Equation 5.2 and given that $\pi(\beta) \in \pi(Tr\,(Proto))$, we have that there exist $tx', ty' \in \mathbb{R}$, $P' \in \mathsf{Agent}$ and $(t, e) \in \beta$ such that $P' = actor\,(e)$, $tx' \leq t \leq ty'$, $(tx', x) \in \beta$, $(ty', y) \in \beta$ and $P \approx P'$. Hence, Lemma 5.2 gives us:

$$ty' - tx' = (ty' - t) + (t - tx') \geq \frac{dist(e,y)}{\mathsf{c}} + \frac{dist(x,e)}{\mathsf{c}} = 2 \cdot \frac{dist(V,P')}{\mathsf{c}},$$

which proves Equation 5.1 given that $(tx', x) \in \beta$, $(ty', y) \in \beta$, $(tz, \mathsf{claim}\,(V, P, x, y)) \in \beta$, and $\beta \sqsubseteq \alpha$. □

The result obtained from Theorem 5.1 means that, within the semantic domain described in Section 5.2.1, the secure distance-bounding property can be verified by simply analyzing the ordering of events in the traces. Therefore, the notions of time and location are indeed unnecessary for the symbolic verification of distance-bounding protocols.

5.4 Conclusions

In this chapter we have studied the symbolic verification framework proposed by Basin et al. [BCSS09, SSBC09]. This framework is based on timed-events and agents' locations and it delivers Isabelle/HOL-constructed (in)security proofs. Motivated by this work, we first characterized a semantic domain of *well-formed* distance-bounding protocols in which the timestamps associated to the agents' actions are only utilized for proximity verification

purposes and not for, e.g., taking a different branch in the execution. This is not a trivial class of distance-bounding protocols but it contains, to the best of our knowledge, all protocols published to date. Later, we proposed our main result, which consists of the first causality-based security model for symbolic verification of distance-bounding protocols, which we prove equivalent to Basin et al.'s model. Our security model can be used to verify any well-formed distance-bounding protocol.

6

Collusion and Terrorist Fraud

Verification of cryptographic protocols is built upon the assumption that participants have not revealed their long-term keys. However, in some protocols participants can collude to defeat some security goals, without revealing their long-term secrets.

In this chapter we develop a multiset rewriting formalism to reason about collusion in security protocols, and introduce the notion of post-collusion security, which verifies security properties claimed in sessions initiated after the collusion. By means of post-collusion security, we extend the distance-bounding security model of Chapter 5 (precisely Theorem 5.1) in order to account for terrorist fraud.

Organization– In Section 6.1, we describe the multiset rewriting model employed by the Tamarin prover. In Section 6.2, we extend this model to formalize the concepts of collusion and post-collusion security. In this section we also show how the notions of collusion post-collusion security apply to authentication analysis. Later on, in Section 6.3, we use the proposed notion of post-collusion security to provide a formal definition of (resistance to) terrorist fraud. A summary of the chapter is provided in Section 7.4.

6.1 Modeling Security Protocols

This section describes the security model we use throughout this and the subsequent chapters. It is based on the multiset rewriting theory employed by the TAMARIN verification tool [MSCB13, SMCB12], which we will use to conduct automatic verification. In the multiset rewriting model, protocols are specified as transition rules and the associated transition system describes the protocol executions. The states of the system are composed of facts. Transition rules model how the protocol participants and the adversary behave and interact.

6.1.1 Preliminaries

Notation. Given a set S, we denote by $S^\sharp$ the set of finite multisets with elements from S, and by S^* the set of finite sequences of elements from S. The power set of S is denoted by $\mathcal{P}(S)$. For any operation on sets, we use the superscript $^\sharp$ to refer to the corresponding operation on multisets, e.g. given the multisets M and M', $M \cup^\sharp M'$ denotes the multiset union of M and M'.

Given a (multi)set S and a sequence $s \in S^*$, $|s|$ denotes the length of s, s_i the i-th element of s with $1 \leq i \leq |s|$, and ε the empty sequence. We write s indistinctly as $[s_1, \ldots, s_n]$ or $s_1 \cdots s_n$ (the choice depends on presentation). The concatenation of the sequences s and s' is denoted by $s \cdot s'$. We use $set(s)$ and $multiset(s)$ to denote the set and multiset of elements from s, respectively. Given $a \in S$ and $s \in S^*$, we write $a \in s$ for $\exists i \in \{1, \ldots, |s|\}.\ a = s_i$.

For the sake of simplicity, when using universal or existential quantifiers, we will not specify the domain unless it is ambiguous or cannot be inferred from the context.

Cryptographic Messages. To model cryptographic messages, we use an order-sorted term algebra $(\mathcal{S}, \leq, \mathcal{T}_\Sigma(\mathcal{V}))$ where $\mathcal{S}$ is a set of sorts, $\leq$ a partial order on $\mathcal{S}$, Σ is a signature, and $\mathcal{V}$ is a countably infinite set of variables. We consider three sorts: $msg, fresh, pub \in \mathcal{S}$, where $fresh \leq msg$ and $pub \leq msg$. That is, msg is the super sort of two incomparable sub-sorts $fresh$ and pub, denoting fresh and public names, respectively. We use $\mathcal{V}_s \subseteq \mathcal{V}$ to denote the set of term variables of sort s and write $x\colon s$ to indicate $x \in \mathcal{V}_s$.

Each function symbol $f \in \Sigma$ has a type $(w, s) \in \mathcal{S}^* \times \mathcal{S}$, where w is the *arity* and s the *sort*. If w is the empty sequence ε, then f denotes a constant of sort s. We use $\Sigma_{w,s} \subseteq \Sigma$ to denote the family of function symbols of type (w, s). Special function families are $\Sigma_{\varepsilon,fresh}$ and $\Sigma_{\varepsilon,pub}$, denoting fresh names (a.k.a. nonces) and public names, respectively. Public names include constants (often written in between quotations, e.g. '`hello`'), and agents' names. The following function symbols are reserved:

- $\langle\ ,\ \rangle \in \Sigma_{msg \times msg, msg}$ to pair two terms.
- $\mathrm{fst}, \mathrm{snd} \in \Sigma_{msg,msg}$ to extract the first and second term from a pair, respectively.
- $\mathrm{senc}, \mathrm{sdec} \in \Sigma_{msg \times msg, msg}$ for symmetric encryption and decryption, respectively. The second argument is the key.
- $\mathrm{aenc}, \mathrm{adec} \in \Sigma_{msg \times msg, msg}$ for asymmetric encryption and decryption, respectively. The second argument is the encryption/decryption key.

- $\mathrm{pk} \in \Sigma_{msg,msg}$ to indicate the asymmetric public key of the argument.
- $\mathrm{sign} \in \Sigma_{msg\times msg,msg}$ and $\mathrm{verify} \in \Sigma_{msg\times msg\times msg,pub}$ to create and verify signatures, respectively.

The semantics of the function symbols above is formalized by the equational theory E, which is in turn defined by the following equations over Σ, where $\mathrm{true} \in \Sigma_{\varepsilon,msg}$:

$$\begin{aligned} &\mathrm{fst}(\langle x, y\rangle) = x, && \mathrm{snd}(\langle x, y\rangle) = y,\\ &\mathrm{sdec}(\mathrm{senc}(x, y), y) = x, && \mathrm{adec}(\mathrm{aenc}(x, \mathrm{pk}(y)), y) = x,\\ &\mathrm{verify}(\mathrm{sign}(x, y), x, \mathrm{pk}(y)) = \mathrm{true}. \end{aligned}$$

We use $t =_E t'$ to denote that terms t and t' are equal modulo E. Terms in our term algebra without free variables are called *ground* terms. A *substitution* is a well-sorted function $\sigma\colon \mathcal{V} \to \mathcal{T}_\Sigma(\mathcal{V})$, i.e. $(\sigma(x) = y \wedge x\colon s) \implies y\colon s$, from variables to terms. We use $t\sigma$ to denote the application of the substitution σ to the term t.

Multiset Rewriting System. We model the execution of a protocol as a labeled transition system. A state in the system is a multiset of *facts*, and a fact is a term of the form $F(t_1, \ldots, t_n)$ where F is a symbol from an unsorted signature Γ and $t_1, \ldots, t_n$ are terms in $\mathcal{T}_\Sigma(\mathcal{V})$. For $n \geq 0$ we denote by $\Gamma^n \subseteq \Gamma$ the set of fact symbols with n arguments. The application of a substitution function σ to a fact $F(t_1, \ldots, t_n)$, denoted $F(t_1, \ldots, t_n)\sigma$, results in the fact $F(t_1\sigma, \ldots, t_n\sigma)$. The set of all facts is denoted $\mathcal{F}$ and the set $\mathcal{G} \subseteq \mathcal{F}$ denotes the set of *ground facts*, which are facts with only ground terms as arguments.

We extend equality modulo E from terms to facts as follows. Given two facts u and u', we write $u =_E u'$ to indicate that they are equal modulo E. Formally, $u =_E u'$ if and only if there exist $n \geq 0$, $t_1, \ldots, t_n, t'_1, \ldots, t'_n \in \mathcal{T}_\Sigma(\mathcal{V})$, and $F \in \Gamma^n$ such that $u = F(t_1, \ldots, t_n)$, $u' = F(t'_1, \ldots, t'_n)$, and $t_i =_E t'_i$ for all $i \in \{1, \ldots, n\}$. Substitution and equality modulo E are also extended to sequences of facts. Given two sequences $s, s' \in \mathcal{F}^*$, $s =_E s'$ if and only if $|s| = |s'|$ and $s_i =_E s'_i$ for all $i \in \{1, \ldots, |s|\}$.

A fact symbol is either *linear* or *persistent*. Linear fact symbols model resources that are exhaustible, such as a message sent to the network. Persistent fact symbols model inexhaustible resources, such as the adversary knowledge. Given a sequence of facts $s \in \mathcal{F}^*$, we write $\mathit{linear}(s)$ and $\mathit{persist}(s)$ to denote the *multiset* of linear facts from s, and the *set* of persistent facts from s, respectively.

We reserve the linear fact symbols $\mathsf{In}, \mathsf{Out}, \mathsf{Fr} \in \Gamma^1$. The facts $\mathsf{In}(m)$ and $\mathsf{Out}(m)$ denote the reception and sending of m, respectively. $\mathsf{Fr}(m)$ indicates that m is a fresh name. The persistent fact symbols $\mathsf{K} \in \Gamma^1$, $\mathsf{Ltk} \in \Gamma^2$, $\mathsf{Shk} \in \Gamma^3$ and $\mathsf{KeyComp} \in \Gamma^1$ are also reserved. $\mathsf{K}(m)$ indicates that the message m is known to the adversary. Facts with symbols Shk and Ltk are used to associate agents to their *long-term* cryptographic keys. $\mathsf{Shk}(A, B, k)$ indicates that k is the long-term symmetric key shared by A and B, and $\mathsf{Ltk}(A, sk)$ indicates that A holds the long-term asymmetric private key sk. We say that an agent A is compromised if the agent has revealed at least one of their long-term keys; and we use the fact $\mathsf{KeyComp}(A)$ to indicate so.

The execution of a protocol starts with the empty multiset of facts, and evolves through *multiset rewriting rules*. A multiset rewriting rule is a tuple (p, a, c), written as $[p] \xrightarrow{a} [c]$, where p, a and c are sequences of facts called the *premises*, the *actions*, and the *conclusions*

of the rule, respectively. Each variable that occurs in a multiset rewriting rule is assumed to be of sort *msg*, unless otherwise indicated.

A *ground instance* of a rule $r := [p] \xrightarrow{a} [c]$ is obtained via application of a substitution function σ to result in $r\sigma := [p\sigma] \xrightarrow{a\sigma} [c\sigma]$ where $p\sigma$, $a\sigma$ and $c\sigma$ consist of ground facts only. Given a set of rules R, we denote $ginsts(R)$ the set of ground instances of the rules from R. We write $g \in_E G$, where g is a (possibly ground) rule and G is a set of (possibly ground) rules, to indicate that $\exists g' \in G.\ g =_E g'$.

A set R of multiset rewriting rules defines a *multiset rewriting system*: an LTS whose set of states is $\mathcal{G}^\sharp$ and whose transition relation $\rightarrow_R \subseteq \mathcal{G}^\sharp \times \mathcal{P}(\mathcal{G}) \times \mathcal{G}^\sharp$ is defined by:

$$
\begin{aligned}
&S \xrightarrow{l}_R S' \iff \\
&\quad \exists (p, a, c) \in_E ginsts(R). \\
&\qquad linear(p) \subseteq^\sharp S \ \wedge\ persist(p) \subseteq set(S) \ \wedge \\
&\qquad S' = (S \setminus^\sharp linear(p)) \cup^\sharp multiset(c) \ \wedge\ l = set(a).
\end{aligned} \tag{6.1}
$$

A transition is performed by applying a ground instance of a transition rule. The rule is applicable if the current system state contains all facts in the premises of the rule. The rule application removes the linear facts from the state, keeps the persistent facts, and adds the facts from the conclusions. An *execution* of R is a finite alternating sequence $[S_0, l_1, S_1, \ldots, l_n, S_n]$ of states and labels such that the following three conditions hold:

(1) $S_0 = \emptyset^\sharp$,

(2) $S_{i-1} \xrightarrow{l_i}_R S_i$ for $1 \leq i \leq n$, and

(3) If $S_{i+1} \setminus^\sharp S_i = \{\mathsf{Fr}(x)\}^\sharp$ for some i and x, then $j \neq i$ does not exist such that $S_{j+1} \setminus^\sharp S_j = \{\mathsf{Fr}(x)\}^\sharp$.

The third condition guarantees that fresh names are generated once. The set of all executions of R is denoted $[\![R]\!]$.

6.1.2 Protocol Specification

A protocol is specified as a set of multiset rewriting rules, called *protocol rules*, with the restrictions: (1) fresh names and K facts do not occur, (2) In and Fr facts do not occur in the conclusions, and (3) every variable occurring in the actions or conclusions either occurs in the premises or is of sort *pub*. The universe of all rules that satisfy these conditions is denoted $\mathcal{R}$.

Example 6.1 (The *Toy* protocol). Figure 6.1 shows a message sequence chart (MSC) [CM12] of the *Toy* protocol, an example protocol which we will use for illustration throughout the paper. The initiator I creates a nonce[1] ni, and sends it to the responder agent R, encrypted with their shared long-term symmetric key. Upon reception, R decrypts the received message to learn ni. Then, R creates his own nonce nr, encrypts it using the nonce ni as a key, and sends that encrypted message to I. Upon reception of $\mathrm{senc}(nr, ni)$,

[1] We will indistinctly use "nonce" and "fresh name", though they mean the same thing: a number generated once.

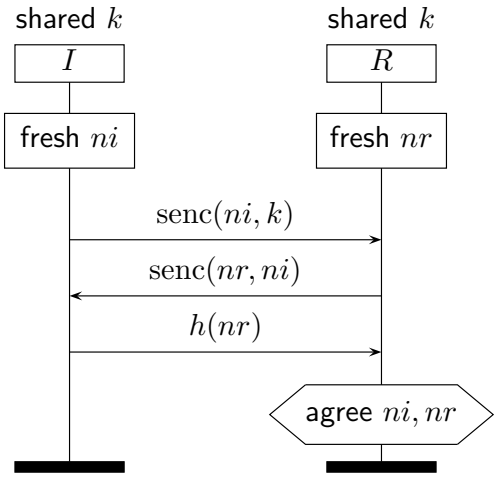


Figure 6.1 The *Toy* protocol.

I learns nr and sends back to R a hash of nr. Such a value allows R to be convinced that I has executed the protocol with R and agrees on the nonces nr and ni. The protocol rules are depicted in Figure 6.2.

The specification of the *Toy* protocol uses fact symbols that are reserved, such as Shk. Indeed, we assume that all protocol specifications use reserved fact symbols with the intended meaning. The remaining facts are used to either determine protocol progress or enrich execution traces with information that will be later used to analyze trace properties.

For example, RState1(I, R, ni, nr) appears in the conclusion of `R1` and in the premises of `R2`, allowing to establish an order between `R1` and `R2`. The facts of the form Start(x) and End(x) denote the start and the end of a protocol run by an agent, respectively. The term x denotes the run identifier. We delay the discussion of the fact symbols Commit and Running until the introduction of security properties in Section 6.1.4.

For the remainder of this chapter, the fact symbols Start and End are reserved to mark the start and end of the protocol execution. Also we assume that the protocol specification is consistent with the usage of Start and End. In particular, we assume that all End facts are reachable from the empty state.

6.1.3 Execution and Adversary Model

Given that protocol rules cannot generate fresh names, i.e. protocol rules are not allowed to use Fr facts in their conclusion, we add a special rule `Fresh`, independent of the protocol specification, to model generation of fresh names. This rule adds a fact Fr(x) into the system state, where x is a variable of type *fresh*, without consuming facts from the state:

$$\texttt{Fresh} := [\,] \rightarrow [\mathsf{Fr}(x)] \text{ where } x\colon \mathit{fresh}.$$

To model the adversary's actions we use the standard Dolev-Yao network adversary, modeled by the rules depicted in Figure 6.3, which we will explain in the next paragraph. We remark that the compromise capabilities of the adversary are part of the protocol specification (e.g. `KeyCompI` and `KeyCompR` in the *Toy* protocol). We will extend this model with collusion actions in the next section.

$$\texttt{KeyGen} := [\mathsf{Fr}(k)] \rightarrow [\mathsf{Shk}(I,R,k)]$$

$$\texttt{ShkCompI} := [\ \mathsf{Shk}(I,R,k)] \xrightarrow{\mathsf{KeyComp}(I)} [\mathsf{Out}(k)]$$

$$\texttt{ShkCompR} := [\ \mathsf{Shk}(I,R,k)] \xrightarrow{\mathsf{KeyComp}(R)} [\mathsf{Out}(k)]$$

$$\texttt{I1} := \begin{bmatrix} \mathsf{Fr}(ni), \\ \mathsf{Shk}(I,R,k) \end{bmatrix} \xrightarrow{\mathsf{Start}(ni)} \begin{bmatrix} \mathsf{Out}(\mathrm{senc}(ni,k)), \\ \mathsf{IState1}(I,R,ni) \end{bmatrix}$$

$$\texttt{R1} := \begin{bmatrix} \mathsf{Fr}(nr), \\ \mathsf{In}(\mathrm{senc}(ni,k)), \\ \mathsf{Shk}(I,R,k) \end{bmatrix} \xrightarrow{\mathsf{Start}(nr)} \begin{bmatrix} \mathsf{Out}(\mathrm{senc}(nr,ni)), \\ \mathsf{RState1}(I,R, \\ ni,nr) \end{bmatrix}$$

$$\texttt{I2} := \begin{bmatrix} \mathsf{IState1}(I,R,ni), \\ \mathsf{In}(\mathrm{senc}(nr,ni)) \end{bmatrix} \xrightarrow{\mathsf{Running}(I,R, \langle \text{`I'}, \text{`R'}, ni, nr \rangle),\ \mathsf{End}(ni)} [\mathsf{Out}\,(h(nr))]$$

$$\texttt{R2} := \begin{bmatrix} \mathsf{RState1}(I,R,ni,nr), \\ \mathsf{In}\,(h(nr)) \end{bmatrix} \xrightarrow{\mathsf{Commit}(R,I, \langle \text{`I'}, \text{`R'}, ni, nr \rangle),\ \mathsf{End}(nr)} [\ ]$$

Figure 6.2 Specification rules of the *Toy* protocol.

$$\texttt{Learn} := [\mathsf{Out}(x)] \rightarrow [\mathsf{K}(x)]$$

$$\texttt{Inject} := [\mathsf{K}(x)] \xrightarrow{\mathsf{K}(x)} [\mathsf{In}(x)]$$

$$\texttt{AdvFresh} := [\mathsf{Fr}(x)] \rightarrow [\mathsf{K}(x)]$$

$$\texttt{Public} := [\ ] \rightarrow [\mathsf{K}(x)] \ \text{ where } x\colon pub$$

$$\texttt{Funct} := [\mathsf{K}(x_1), \ldots, \mathsf{K}(x_n)] \rightarrow [\mathsf{K}\,(f(x_1, \ldots, x_n))]$$

Figure 6.3 Dolev-Yao rules.

The rules `Learn` and `Inject` model the adversary's ability to learn messages being sent and to inject any of their known messages, respectively. The rule `AdvFresh` declares that the adversary can generate their own fresh names. The rule `Public` states that the adversary knows all public messages and the rule `Funct` indicates that the adversary can evaluate any function, provided that they know the inputs.

We denote by $\mathcal{I}$ the set of intruder rules in Figure 6.3 together with the rule `Fresh`, which will form part of every protocol specification. Hence, the set of all executions of a set *Proto* of protocol rules is $[\![Proto \cup \mathcal{I}]\!]$. Moreover, given an execution $[S_0, l_1, S_1, \ldots, l_n, S_n]$ of *Proto*, the sequence $l_1 \cdots l_n$ is called the *trace*. The set of all traces of *Proto* is denoted $Tr\,(Proto)$, i.e $Tr\,(Proto) = \{l_1 \cdots l_n \mid [S_0, l_1, S_1, \ldots, l_n, S_n] \in [\![Proto \cup \mathcal{I}]\!]\}$.

6.1.4 Security Properties

Security properties are verified on execution traces. Certain facts on the traces indicate a security *claim*, e.g. the Commit fact in the *Toy* protocol. A security claim denotes a belief

(traditionally of an agent) about the protocol execution that led to the claim. Formally, we define a security property φ as a relation on traces and integer numbers such that $\varphi(t_1 \cdots t_n, i)$ means that security claims in t_i are valid. Recall that a trace is a sequence of labels, which in turn are sets of ground facts.

For illustration purposes, let us instantiate φ with the authentication property *non-injective agreement*, as defined by Lowe in [Low97]. Following Lowe's notation, we use the fact symbols Running and Commit as markers in the traces to indicate those execution steps where agreement is expected to be satisfied, e.g. as used by rules `I2` and `R2` in Figure 6.2.

Non-injective agreement on a message m is guaranteed to an agent A, if whenever A completes a run apparently with B, denoted by the claim $\mathsf{Commit}(A, B, m)$, then B has previously performing a run apparently with A and they both agree on m, denoted by the fact $\mathsf{Running}(A, B, m)$:

$$\begin{aligned} &ni\_agree(t, i) \iff \\ &\quad \forall A, B, m.\ \mathsf{Commit}(A, B, m) \in t_i \implies \\ &\qquad (\exists j.\ \mathsf{Running}(B, A, m) \in t_j) \vee \\ &\qquad (\exists j.\ \mathsf{KeyComp}(A) \in t_j \vee \mathsf{KeyComp}(B) \in t_j)\,. \end{aligned} \tag{6.2}$$

On the one hand, the prefix-closure of traces imposes an implicit order $j < i$ in Equation 6.2, which is suggested by the word "previously" in the description of the property. On the other hand, the last line of Equation 6.2 indicates that the property is conditional on A and B not being compromised.

Definition 6.1 (Security). A set *Proto* of protocol rules *satisfies* a security property φ, denoted $Proto \models \varphi$, if:

$$\forall t \in Tr\,(Proto)\,, i \in \{1, \ldots, |t|\}.\ \varphi(t, i).$$

The *Toy* protocol satisfies[2] non-injective agreement, i.e. $Toy \models ni\_agree$. We write $Proto \not\models \varphi$ to indicate that $Proto \models \varphi$ does not hold.

6.2 Collusion

In recent years, the adversary model of Dolev and Yao [DY83] has become an accepted standard. The Dolev-Yao adversary is capable of intercepting, blocking or modifying messages on the communication network, as well as injecting their own messages. Further, the adversary is assumed to be capable of *compromising* protocol participants (a.k.a. users, agents), gaining full control of them for the entire protocol execution. Indeed, computer-assisted verification approaches typically consider Dolev-Yao adversaries. Such approaches have proven useful in verifying or discovering attacks on real-world, complex protocols such as 5G authentication [BDH+18], the TLS 1.3 protocol suite [CHH+17], and key-exchange protocols such as Naxos [SMCB12].

In some cases, the Dolev-Yao model has been shown to be too coarse-grained. This is because this model assumes that agents can be categorized as being either honest: those

[2] All Tamarin models and proofs used for this thesis can be found in our repository https://github.com/jorgetp/dbverify.

who precisely follow their protocol specification; or compromised: those who deviate from their protocol specification as desired by the adversary. The concern lies in accounting for agents who cannot be classified in either group.

For example, a given agent might choose to deviate from the protocol specification, but only if certain guarantees are met in later executions of the protocol. Would a university student willingly, due to certain benefit, lend their campus access card to a university-external friend? The student's decision might be conditional on their assertion that their friend will not be able to later access the campus, after the card has been returned to its owner. Would a user of a video streaming platform utilize a VPN extension to fool geo-location restrictions? The user's decision might be based on whether they are certain that the VPN extension is not malicious and will not cause irreversible harm.

In this section we refine the traditional Dolev-Yao adversary model in order to capture *collusion*. Collusion refers to any *deviation* of the protocol specification by agents who are not under control of the adversary. Furthermore, we introduce the notion of *post-collusion security*, which refers to security guarantees about claims made in execution sessions initiated after the collusion. Informally, one can interpret the relation of these two notions as follows: post-collusion security allows the potential colluding agents to decide whether colluding is worth it. After all, what the agents gain out of colluding must outweigh the collateral effect that such collusion might have on themselves. On the other hand, a protocol designer might aim to increase the cost of collusion.

More precisely, in Section 6.2.1 we extend the adversary model with collusion rules, which express ways in which non-compromised agents can deviate from the protocol specification. In an illustrative example, we show how such extension invalidates, under the traditional security verification model (Definition 6.1), the agreement-based authentication property given earlier (Section 6.1.4). Later, in Section 6.2.2 we provide the formulation of post-collusion security.

6.2.1 Collusion Rules

In the traditional Dolev-Yao compromise model, agents are assumed to be either *compromised* (a.k.a. corrupt, dishonest) or *non-compromised* (a.k.a. honest). Non-compromised agents follow precisely the protocol specification, whilst compromised agents deviate from it as pleased by the adversary.

We refine the traditional Dolev-Yao compromise model so that agents can *collude* in order to provide false proof to their communication partners of a certain claim's validity. Collusion refers to non-compromised agents' deviation from their protocol specification.

For example, assume *Alice* is running an authentication protocol (supposedly) with *Bob*. Consider also a third party *Charlie* who, in cooperation with *Bob*, impersonates *Bob* when communicating with *Alice*. *Bob* could trivially achieve this by giving all his secret keys to *Charlie*. But, does *Bob* really have to do so in order to cheat on *Alice*? Not necessarily. Indeed, *Bob* can provide *Charlie* (possibly in advance) with all the messages that *Charlie* needs to successfully complete a protocol session with *Alice*, posing as *Bob*. Such aid by *Bob* is what we call collusion, and we call *Bob* a colluding agent.

Collusion is modelled by extending the protocol specification such that agents may deviate from the protocol's intended execution. Typically, the deviation consists of leakage of session data, cryptographic oracles, reuse of nonces, or state reveals. For example, in the

Toy protocol I might collude with a compromised agent, say *Eve*, by leaking ni. This can be modelled with the rule:

$$\texttt{Leak\_ni} := \left[\mathsf{IState1}(I, R, ni)\right] \xrightarrow{\mathsf{Collusion}(\,)} \begin{bmatrix}\mathsf{IState1}(I, R, ni),\\ \mathsf{Out}(ni)\end{bmatrix} \tag{6.3}$$

which leads to the statement:

$$\mathit{Toy} \;\cup\; \{\texttt{Leak\_ni}\} \not\models \mathit{ni\_agree}\ .$$

The rules that extend the protocol specification to model collusion are called *collusion rules*. By convention, and also to syntactically distinguish legitimate protocol rules from collusion rules, we will assume that all collusion rules have an action fact of the form $\mathsf{Collusion}(\,)$. We denote by $\mathcal{C} \subseteq \mathcal{R}$ the universe of all collusion rules. We restrict the set of collusion rules by requiring them to not prevent agents from completing legitimate protocol runs.

Definition 6.2 (Valid Extension)**.** Let $\mathit{Proto} \subseteq \mathcal{R} \setminus \mathcal{C}$ be a protocol and $C \subseteq \mathcal{C}$ be a set of collusion rules, we say that $\mathit{Proto}' = \mathit{Proto} \cup C$ is a *valid extension* of Proto if:

$$\begin{aligned}&\forall \alpha \in \mathit{Tr}\left(\mathit{Proto}'\right), i, x.\\ &\quad (\mathsf{Start}(x) \in \alpha_i \wedge \nexists j.\, \mathsf{End}(x) \in \alpha_j) \implies\\ &\qquad \exists \beta.\, \alpha \cdot \beta \in \mathit{Tr}\left(\mathit{Proto}'\right) \wedge \mathsf{End}(x) \in \beta_{|\beta|}.\end{aligned}$$

Definition 6.2 states that collusion rules do not create points of no-return during execution. That is to say, agents must always be able to complete their runs even if they have colluded. For example, $\mathit{Toy} \cup \{\texttt{Leak\_ni}\}$ is a valid extension of Toy because, even if I leaks ni in some execution, I might still continue with the intended protocol execution. This means that the system can execute the next rule concerning I, i.e. `I2`, should the system state contain the facts $\mathsf{IState}(I, R, ni)$ and $\mathsf{In}(\mathrm{senc}(ni, nr))$ for some nr. Thus, I can log the fact $\mathsf{Running}(I, R, \langle \text{`}\texttt{R}\text{'}, \text{`}\texttt{I}\text{'}, nr, ni\rangle)$ into the trace which means that indeed I has been running the protocol with R and they both agree on ni and nr. This is particularly important for authentication properties. After all, they are meant to guarantee an agent's claim about their peer's local state.

An example of a rule that leads to a non-valid extension of *Toy* is the following:

$$\texttt{NonValidRule} := \left[\mathsf{IState1}(I, R, ni)\right] \xrightarrow{\mathsf{Collusion}()} [\,]\ .$$

Thus $\mathit{Toy} \cup \{\texttt{NonValidRule}\}$ is not a valid extension of *Toy* because if this rule is applied, it will consume the fact $\mathsf{IState1}(I, R, ni)$ from the system state, and so I will never be able to continue with the current run (identified by ni).

6.2.2 Post-Collusion Security

We informally define post-collusion security as follows.

Definition 6.3 (Informal)**.** Post-collusion security is the guarantee of security claims made in sessions initiated *after* the collusion.

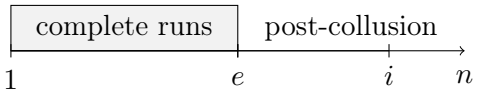


Figure 6.4 A trace $t = t_1 \cdots t_e \cdots t_i \cdots t_n$ can be broken down into a pre-collusion trace consisting of completed runs (e.g. before e), and a second subtrace containing post-collusion claims (e.g. a claim made in t_i).

The remainder of this section is intended to formalise the above informal definition of post-collusion security. Moreover, we will use the *Toy* protocol and the agreement property from the previous sections to illustrate our definitions and intuitions.

To identify claims made in sessions initiated after the collusion, which we call *post-collusion claims*, we must make sure that all sessions before or while the (last) collusion occurred are complete. The reason for this is that the agent who makes a security claim cannot always decide whether their communication partner is still acting on a run initiated before or during the collusion. That is, a claim by *Alice* about her communication with *Bob* is a post-collusion claim if both *Alice* and *Bob* have completed their runs that started before or while *Bob* colluded. That way we make sure that *Alice* makes her claim in a session initiated after *Bob*'s collusion action.

Consider a trace $t = t_1 \cdots t_e \cdots t_i \cdots t_n$, and an index e such that all collusion actions (if any) occurred before e. If all runs initiated before e were completed before e too, then we call the security claims made after e post-collusion claims. See Figure 6.4 for a graphical representation. Note that every claim that occurs after a post-collusion claim is also a post-collusion claim.

Below in Definition 6.4 we formulate post-collusion security, in which we use the following helper predicates on sequences of sets of ground facts:

$$\begin{aligned} &\mathit{complete}(l) \iff \forall i, x.\ (\mathsf{Start}(x) \in l_i \implies \exists j.\ \mathsf{End}(x) \in l_j), \\ &\mathit{nocollusion}(l) \iff \nexists j.\ \mathsf{Collusion}(\,) \in l_j. \end{aligned}$$

On the one hand, $\mathit{complete}(l)$ holds if all runs initiated in l are also completed in l. On the other hand, $\mathit{nocollusion}(l)$ means that no collusion actions occurred in l.

Definition 6.4 (Post-Collusion Security). Given a protocol *Proto*, a valid extension *Proto*′ of *Proto*, and a security property φ, we say that *Proto*′ *is post-collusion secure with respect to* φ, denoted $\mathit{Proto}' \models^\star \varphi$, if:

$$\begin{aligned} &\forall t \in \mathit{Tr}\left(\mathit{Proto}'\right), e \in \{1, \ldots, |t|\}. \\ &\quad (\mathit{complete}(t_1 \cdots t_e) \wedge \mathit{nocollusion}(t_{e+1} \cdots t_{|t|})) \\ &\qquad \implies \forall i > e.\ \varphi(t, i). \end{aligned} \tag{6.4}$$

We write $\mathit{Proto}' \not\models^\star \varphi$ to indicate that $\mathit{Proto}' \models^\star \varphi$ does not hold. As Figure 6.5 shows, $\mathit{Toy} \cup \{\texttt{Leak\_ni}\}$ is *not* post-collusion secure with respect to non-injective agreement, i.e.

$$\mathit{Toy} \cup \{\texttt{Leak\_ni}\} \not\models^\star \mathit{ni\_agree}. \tag{6.5}$$

The attack works with two consecutive sessions in which an attacker *Eve* can re-use the first message $\mathrm{senc}(ni, k)$ and ni of the first session to impersonate I in the second session.

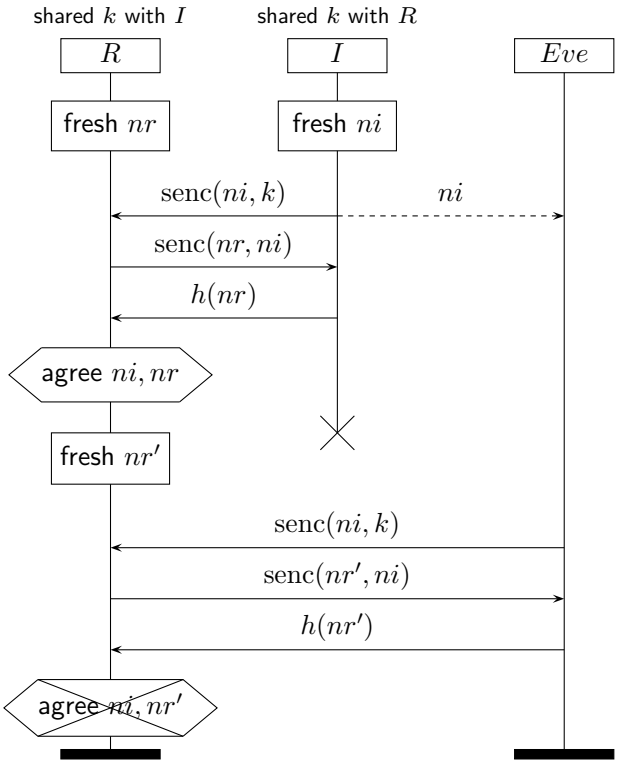


Figure 6.5 An MSC showing that the *Toy* protocol with collusion, represented by the dashed arrow, is not post-collusion secure with respect to non-injective agreement.

Observe that the second claim is a post-collusion claim as the first session is complete and no collusion occurred in the second session.

The impact of post-collusion security can depend on the circumstances in which a given protocol is deployed. We see from the *Toy* protocol that the effects of collusion can cause an irreversible change to the truth value of future authentication claims. Thus, a legitimate agent playing the initiator role would not want to collude with a "friend" by giving them their nonce ni, as this would lead to impersonation. On the contrary, suppose a given protocol is post-collusion secure with respect to a desirable authentication property. Then, an agent can issue their "one-time" keys to their friends if desired, confident that these friends will not be able to re-use this information for later authentication.

6.3 Post-Collusion Security in Distance Bounding

In this section we use post-collusion security to develop a symbolic formulation of (resistance to) terrorist fraud in distance-bounding protocols. First we give a protocol example in order to illustrate the modeling of distance-bounding protocols by using the multiset rewriting model from Section 6.1. We pay particular attention to restrictions to the Dolev-Yao model that are necessary to model physical limitations of the communication channel. Later on, in Section 6.3.1, we formulate the *secure distance-bounding* property proposed in [MSTT18b] to verify this type of protocols. Finally, in Section 6.3.2, we provide a symbolic formulation of resistance to terrorist fraud.

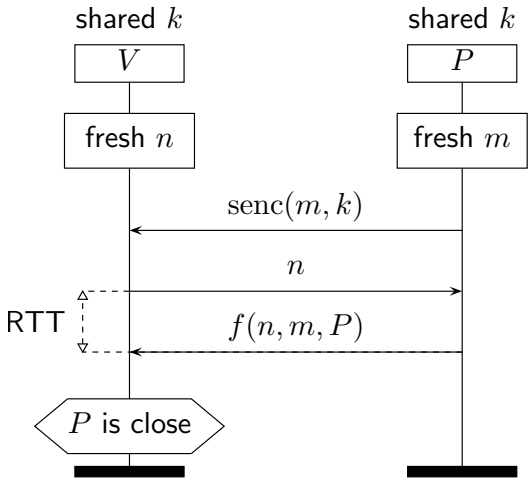


Figure 6.6 The *DBToy* protocol.

Example 6.2 (The *DBToy* Protocol)**.** Figure 6.6 depicts the *DBToy* protocol, which works as follows. The prover P encrypts a fresh name m with the shared key between P and the verifier V. Then P sends the encrypted message to V. Hence, the fast phase starts with V sending the fresh name n as the challenge, to which P must reply with $f(n, m, P)$. If P replies correctly and on time, then V declares P as being close. The specification rules of *DBToy* are shown in Figure 6.7.

In the rules we introduced the linear fact symbols $\mathsf{Send}, \mathsf{Recv} \in \Gamma^2$, $\mathsf{Action} \in \Gamma^1$ and $\mathsf{DBSec} \in \Gamma^4$. A fact $\mathsf{Send}(X, m)$ denotes the sending of m by the agent X and a fact $\mathsf{Recv}(X, m)$ denotes the reception by X of the message m. A fact $\mathsf{Action}(X)$ denotes that an action was executed by X. A fact $\mathsf{DBSec}(V, P, ch, rp)$ denotes V's claim that P is close during the fast phase delimited by $\mathsf{Send}(V, ch)$ and $\mathsf{Recv}(V, rp)$. The remaining newly introduced facts denote the agents' information on the system state. Recall that the (reserved) fact symbols Shk and $\mathsf{KeyComp}$ are persistent. The rest of the fact symbols used in Figure 6.7 are linear.

The rules `DBNet` and `DBAdv` were introduced in [MSTT18b] to restrict the Dolev-Yao attackers' communication with protocol participants. This was briefly motivated in Section 2.2. The aim is to capture the statement "every message that can be received by the verifier during the fast phase has been sent from a real physical location". The reason behind this is that messages cannot travel faster than light, thus the adversary cannot instantaneously send a message to an agent (as modelled by Dolev-Yao's rule `Inject` in Figure 6.3). Hence, in order for the adversary to inject data on an agent's receiver, they must use a compromised agent as the sender, ergo the message takes a while to arrive to the receiver. Note that we do not drop the rule `Inject` but we use Recv and Send facts to model the sending and receiving of messages during the fast phase.

In line with this, we will assume that every set of rules defining a distance-bounding protocol is consistent with the usage of Send, Recv and Action facts as follows: (1) every message m sent by the prover P is modeled by a rule with a fact $\mathsf{Send}(P, m)$ in the conclusions, (2) every message m received by the verifier V *during the fast phase* is modeled by a rule with a fact $\mathsf{Recv}(V, m)$ in the premises, and (3) every prover rule has the action

$$\texttt{KeyGen} := [\mathsf{Fr}(k)] \rightarrow [\mathsf{Shk}(V,P,k)]$$

$$\texttt{KeyRevV} := [\mathsf{Shk}(V,P,k)] \xrightarrow{\mathsf{KeyComp}(V)} \begin{bmatrix}\mathsf{Out}(k),\\ \mathsf{KeyComp}(V)\end{bmatrix}$$

$$\texttt{KeyRevP} := [\mathsf{Shk}(V,P,k)] \xrightarrow{\mathsf{KeyComp}(P)} \begin{bmatrix}\mathsf{Out}(k),\\ \mathsf{KeyComp}(P)\end{bmatrix}$$

$$\texttt{DBNet} := [\mathsf{Send}(X,m)] \xrightarrow{\mathsf{Action}(Y),\ \mathsf{Recv}(Y,m)} \begin{bmatrix}\mathsf{Out}(m),\\ \mathsf{Recv}(Y,m)\end{bmatrix}$$

$$\texttt{DBAdv} := \begin{bmatrix}\mathsf{In}(m),\\ \mathsf{KeyComp}(X)\end{bmatrix} \xrightarrow{\mathsf{Action}(X)} [\mathsf{Send}(X,m)]$$

$$\texttt{P1} := \begin{bmatrix}\mathsf{Fr}(m),\\ \mathsf{Shk}(V,P,k)\end{bmatrix} \xrightarrow{\mathsf{Start}(m),\ \mathsf{Action}(P)} \begin{bmatrix}\mathsf{Send}(P,\mathrm{senc}(m,k)),\\ \mathsf{ProvSt1}(P,m)\end{bmatrix}$$

$$\texttt{V1} := \begin{bmatrix}\mathsf{Fr}(n),\\ \mathsf{Shk}(V,P,k),\\ \mathsf{In}(\mathrm{senc}(m,k))\end{bmatrix} \xrightarrow[\mathsf{Send}(V,n)]{\mathsf{Start}(n),} [\mathsf{VerifSt1}(V,P,n,m)]$$

$$\texttt{P2} := \begin{bmatrix}\mathsf{ProvSt1}(P,m),\\ \mathsf{In}(n)\end{bmatrix} \xrightarrow{\mathsf{Action}(P),\ \mathsf{End}(m)} [\mathsf{Send}(P,f(n,m,P))]$$

$$\texttt{V2} := \begin{bmatrix}\mathsf{VerifSt1}(V,P,n,m),\\ \mathsf{Recv}(V,f(n,m,P))\end{bmatrix} \xrightarrow[\mathsf{End}(n)]{\mathsf{DBSec}(V,P,n,f(n,m,P)),} [\,]$$

Figure 6.7 Specification rules of the *DBToy* protocol.

fact $\mathsf{Action}(P)$ where P is the prover's name.

6.3.1 Secure Distance-Bounding

In Definition 5.2, we formulated a causality-based property to verify distance-bounding protocols. The property resembles a form of *aliveness* [Low97, CM12] as the prover must perform some action during the fast phase of the protocol. The authors demonstrated that a verifier's guarantee that the prover is alive during the fast phase is equivalent to the verifier's guarantee that the fast phase RTT provides an upper bound to their distance to the prover. Next we formulate the property:

$$\begin{aligned}
&dbsec(t,l) \iff\\
&\quad \forall V,P,ch,rp.\ \mathsf{DBSec}(V,P,ch,rp) \in t_l \implies\\
&\qquad (\exists i,j,k.\ i<j<k \wedge \mathsf{Send}(V,ch) \in t_i \wedge\\
&\qquad\quad \mathsf{Action}(P) \in t_j \wedge \mathsf{Recv}(V,rp) \in t_k) \vee\\
&\qquad (\exists b,b',i,j,k,P'.\\
&\qquad\quad i<j<k \wedge \mathsf{Send}(V,ch) \in t_i \wedge\\
&\qquad\quad \mathsf{Action}(P') \in t_j \wedge \mathsf{Recv}(V,rp) \in t_k \wedge\\
&\qquad\quad \mathsf{KeyComp}(P) \in t_b \wedge \mathsf{KeyComp}(P') \in t_{b'}) \vee\\
&\qquad (\exists i.\ \mathsf{KeyComp}(V) \in t_i).
\end{aligned} \tag{6.6}$$

Secure distance-bounding holds for a trace t if, whenever a claim $\mathsf{DBSec}(V, P, ch, rp)$ occurs, it is the case that there is an action of P (or a compromised prover P' if P is compromised) during the fast phase. TAMARIN provides proof of $DBToy \models dbsec$.

Observe that, unlike the agreement property from Section 6.1, *dbsec* does not exclude traces in which one of the agents involved in the security claim is compromised. Instead, should the prover be compromised, then the verification fails only if no compromised prover is active in the fast phase.

6.3.2 Formalizing (Resistance To) Terrorist Fraud

In this section we provide a formal, symbolic definition of resistance to terrorist fraud. Recall that this is an attack in which agents collude to falsely prove proximity for one run of the protocol, whereas no further false proximity proofs can be issued without further collusion. We informally define this attack as follows.

Definition 6.5 (Informal). Terrorist fraud (TF) is an attack in which a remote and non-compromised prover P colludes with a close and compromised prover P' to make the verifier believe that P is close. Conditionally, P' (or any other compromised prover) must not be able to attack the protocol again without further collusion.

While the *dbsec* property allows us to detect attacks such as distance fraud [Des88] and distance hijacking [CRSC12], it is too fine-grained for modelling terrorist fraud, as we require the distant and colluding prover to be non-compromised. In line with this reasoning, we define below a property broader than *dbsec*, that is conditional on non-compromise of both prover and verifier:

$$
\begin{aligned}
&dbsec\_hnst(t, l) \iff \\
&\quad \forall V, P, ch, rp.\ \mathsf{DBSec}(V, P, ch, rp) \in t_l \implies \\
&\qquad (\exists i, j, k.\ i < j < k \wedge \mathsf{Send}(V, ch) \in t_i \wedge \\
&\qquad\qquad \mathsf{Action}(P) \in t_j \wedge \mathsf{Recv}(V, rp) \in t_k) \vee \\
&\qquad (\exists i.\ \mathsf{KeyComp}(V) \in t_i \vee \mathsf{KeyComp}(P) \in t_i).
\end{aligned}
$$

We formally define next *resistance to terrorist fraud*, a property formulated by means of post-collusion security with respect to *dbsec_hnst*.

Definition 6.6 (Resistance to terrorist fraud). A protocol $Proto \subseteq \mathcal{R} \setminus \mathcal{C}$ is *resistant to terrorist fraud* if every valid extension $Proto'$ of $Proto$ that breaks *dbsec_hnst* is *not* post-collusion secure with respect to *dbsec_hnst*, i.e.

$$
Proto' \not\models dbsec\_hnst \implies Proto' \not\models^{\star} dbsec\_hnst. \tag{6.7}
$$

Observe that resistance to terrorist fraud is a property on protocols rather than on traces. Further, note that terrorist fraud uses the negation of post-collusion security. That is because, in a terrorist fraud attack, the colluding prover wishes to allow their partner to complete the protocol only whilst they are cooperating.

Theorem 6.1. *DBToy* is resistant to terrorist fraud.

Proof. Let $DBToy'$ be a valid extension of $DBToy$ such that $DBToy' \not\models dbsec\_hnst$. Thus, there exist $t_1 \cdots t_l \in Tr\,(DBToy')$, and $n, m, V, P \in \mathcal{T}_\Sigma$, and $i, k \in \{1, \ldots, l\}$ with $i < k$, such that:

$$\begin{aligned}
&\mathsf{Send}(V, n) \in t_i \wedge \mathsf{Recv}(V, f(n, m, P)) \in t_k \wedge \\
&\mathsf{DBSec}(V, P, n, f(n, m, P)) \in t_l \wedge \\
&(\nexists j \in \{i+1, \ldots, k-1\}.\ \mathsf{Action}(P) \in t_j) \wedge \\
&(\nexists j \in \{1, \ldots, l\}.\ \mathsf{KeyComp}(V) \in t_j) \wedge \\
&(\nexists j \in \{1, \ldots, l\}.\ \mathsf{KeyComp}(P) \in t_j),
\end{aligned} \tag{6.8}$$

Hence, because of Equation 6.8 above and given that the fact $\mathsf{Recv}(V, f(n, m, P))$ can only occur due to the rule `DBNet` (see Figure 6.7), we derive that:

$$\exists c, j \in \{1, \ldots, k-1\}, C.\ (\mathsf{Send}(C, f(n, m, P)) \in t_j \wedge \mathsf{KeyComp}(C) \in t_c)\,. \tag{6.9}$$

Equation 6.9 implies that $\exists w < k.\ \mathsf{K}(m) \in t_w$. This means that $DBToy'$ has a collusion rule in which m is given away. In addition, if the adversary knows m, then they can use a compromised prover to run again the protocol with V on behalf of P, by using the messages $\mathrm{senc}(m, k)$ and $f(n_2, m, P)$ in that order, where n_2 is V's (new) challenge. This reasoning can be formalized as follows.

Given that $DBToy'$ is a valid extension of $DBToy$ (see Definition 6.2) we have that $e \geq l$, and $t_{l+1}, \ldots, t_e$ exist such that:

$$t_1 \cdots t_l \cdots t_e \in Tr\left(DBToy'\right) \ \wedge \ complete(t_1 \cdots t_l \cdots t_e). \tag{6.10}$$

Now, $l_2 \geq e$, and $t_{e+1}, \ldots, t_{l_2}$, and n_2, and $i_2, k_2 \in \{e+1, \ldots, l_2 - 1\}$ exist such that:

$$\begin{aligned}
&t_1 \cdots t_l \cdots t_e \cdots t_{l_2} \in Tr\left(DBToy'\right) \ \wedge \\
&\mathsf{Send}(V, n_2) \in t_{i_2} \wedge \mathsf{Recv}(V, f(n_2, m, P)) \in t_{k_2} \wedge \\
&\mathsf{DBSec}(V, P, n_2, f(n_2, m, P)) \in t_{l_2} \wedge \\
&(\nexists j \in \{i_2+1, \ldots, k_2-1\}.\ \mathsf{Action}(P) \in t_j) \wedge \\
&(\nexists j \in \{1, \ldots, l_2\}.\ \mathsf{KeyComp}(V) \in t_j) \wedge \\
&(\nexists j \in \{1, \ldots, l_2\}.\ \mathsf{KeyComp}(P) \in t_j).
\end{aligned} \tag{6.11}$$

Therefore, from Equations 6.10 and 6.11 we deduce that $DBToy' \not\models^\star dbsec\_hnst$[3] which completes the proof. □

6.4 Conclusions

In this chapter we briefly discussed about security protocols in the presence of colluding agents. Colluding agents are agents who are not under full control of the adversary, yet they are willing to deviate from the intended protocol execution with the goal to invalidate a given property. Agents can collude to break a given security property, i.e. make a proving participant believe that certain properties hold, even if the proving party's communication

[3] A Tamarin proof for a given $DBToy'$ is also available in our repository.

partners are not under full control of the adversary. We introduced the notion of post-collusion security, which provides security guarantees even if the agents involved have been colluding in previous sessions of the protocol.

We have proposed a concrete symbolic formulation of post-collusion security that can be implemented in state-of-the-art protocol verification tools such as Tamarin. We used our definition to illustrate that leakage of session data can lead to impersonation of agents. This is particularly interesting in the context of authentication properties in which agents, by leaking session-fresh data that is apparently harmless for later executions, enable the adversary to successfully break the authentication property in every session thereafter.

By means of post-collusion security, we provided a symbolic definition of (resistance to) terrorist fraud. Recall that this is an attack in which a distant prover colludes with a compromised prover who is co-located with the verifier. As a result, they cause the verifier to falsely believe that the distant prover is proximal; whilst no further false proximity proof can be provided without further collusion.

7

Automatic Verification

In this chapter we show how our definitions from Chapters 5 and 6 can be used to automatically deliver symbolic proofs of (in)security of distance-bounding protocols. Specifically, we utilize the distance-bounding security model of Chapter 5, its collusion-based extension developed in Chapter 6, and the Tamarin *prover to build a verification framework that accounts for all four classes of distance-bounding attacks: mafia fraud, distance fraud, distance hijacking, and terrorist fraud.*

By using our framework, we conduct a comprehensive security survey of protocols, including industrial protocols such as EMV contactless payment protocols. We provide fixes for the vulnerabilities encountered in the industrial protocols and provide computer-verifiable security proofs of the repaired protocols.

Organization– In order to illustrate the overall verification methodology we employ, we perform a detailed analysis of the TREAD protocol [ABG+17] in Section 7.1. In Section 7.2, we describe the results of our verification of a selection of over 25 distance-bounding protocols. In this section we also discuss differences between our verification results and those of Chothia et al.'s [CdRS18]. Finally, we analyze in detail the industrial protocols in Section 7.3 based on the ISO/IEC 14443 standard (includes EMV payment protocols) and propose fixes for the vulnerabilities encountered. We summarize our findings in Section 7.4.

7.1 Breaking the TREAD Protocol

A Prover-Anonymous and Terrorist-Fraud Resistant Distance-Bounding Protocol (TREAD) is shown in Figure 7.1 and it consists of the traditional three phases. First, the prover P generates two nonces α and β, and creates the message $\langle \alpha, \beta, \mathsf{idpriv}(P) \rangle$, where $\mathsf{idpriv}(P)$ is an anonymous group identity. This message is signed by P, then encrypted, and sent to the verifier V, together with P's public identity $\mathsf{idpub}(P)$. Upon reception, V decrypts the message and verifies the signature. If correct, V finishes the first phase by sending a random nonce m of size n to P. The second phase is a standard n-round fast phase wherein V sends a random bit c_i and P replies back with α_i if $c_i = 0$, with $\beta_i \oplus m_i$ otherwise. The protocol finishes successfully if all responses during the fast phase are correct and the round-trip times are below a predefined threshold (third phase).

The TREAD protocol was claimed to satisfy various security properties, in the computational model [DFKO11]. Relying on this model, a proof is given to show probabilistic resistance[1] against mafia fraud, distance fraud, terrorist fraud, and distance hijacking attacks. However, by using our framework, we have identified mafia fraud and distance hijacking attacks on this protocol.

To symbolically verify this protocol, we transform the fast phase into a single challenge response message exchange (see Figure 7.2). We also ignore details that are irrelevant to our security analysis, such as the anonymous identity of the prover, and upgrade bitwise operations to stronger cryptographic primitives, such as a hash function. Overall, our goal is to obtain an abstraction of the original protocol such that every attack found in the abstraction can be mapped back onto the original protocol.

The TREAD can be instantiated with either a asymmetric or symmetric encryption. We thus specified it in Tamarin with both variants: one where k is an asymmetric key and another one where k is symmetric. In the first variant, Tamarin finds a simple man-in-the-middle attack that violates the secure distance-bounding property Recall the property from Definition 5.2 and formulated in the multiset rewriting language in Equation 6.6. The attack is a mafia fraud and is depicted in Figure 7.3. A prover P starts a session with a dishonest verifier E by sending message $\langle \{\alpha, \beta, P, \{\alpha, \beta, P\}_{sk(P)}\}_{pk(E)}, P \rangle$ to E. Hence, E decrypts the received message, learns the nonces α and β, and re-encrypts the message with the public key of a legitimate verifier V. Next, the intruder starts a session with V with the goal of impersonating P. To do so, E sends $\langle \{\alpha, \beta, P, \{\alpha, \beta, P\}_{sk(P)}\}_{pk(V)}, P \rangle$ to V. Then V checks that the signed message $\{\alpha, \beta, P\}_{sk(P)}$ indeed corresponds to P, and sends back two nonces m and c, where c is the fast phase challenge. The attack ends with the intruder correctly replying to the challenge c with $f(c, m, \alpha, \beta)$.

The attack described above not only breaks standard authentication properties such as agreement and synchronization [Low97, CM12], but also the secure distance-bounding property as follows. Assume P is far from V and the intruder E wants to convince V that P is close. To achieve this, E needs to be close to V and executes the attack above. Note that the fast phase corresponds to the events containing the messages c and $f(c, m, \alpha, \beta)$, which the intruder can successfully produce without relaying.

Interesting enough, if k is a symmetric key the described mafia fraud attack no longer works. The reason is that the intruder does not know the secret key shared between P and

[1] No attack succeeds with non-negligible probability.

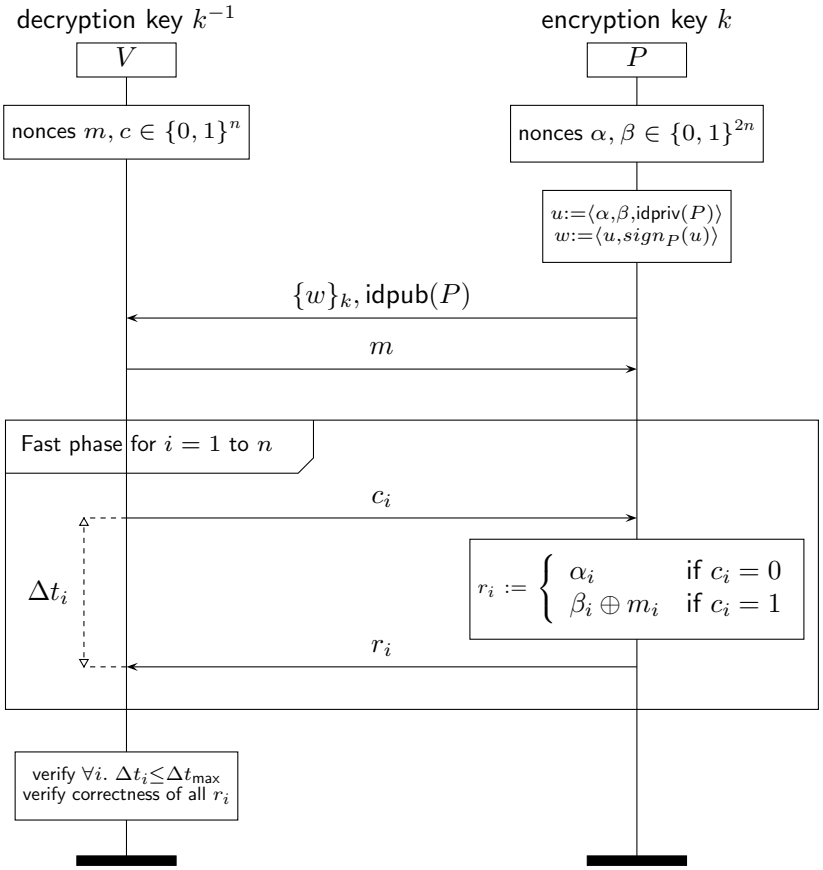


Figure 7.1 The TREAD protocol.

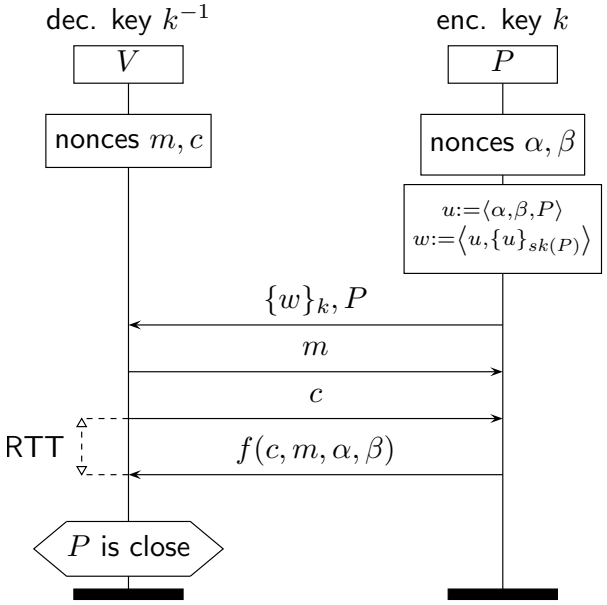


Figure 7.2 A symbolic abstraction of the TREAD protocol.

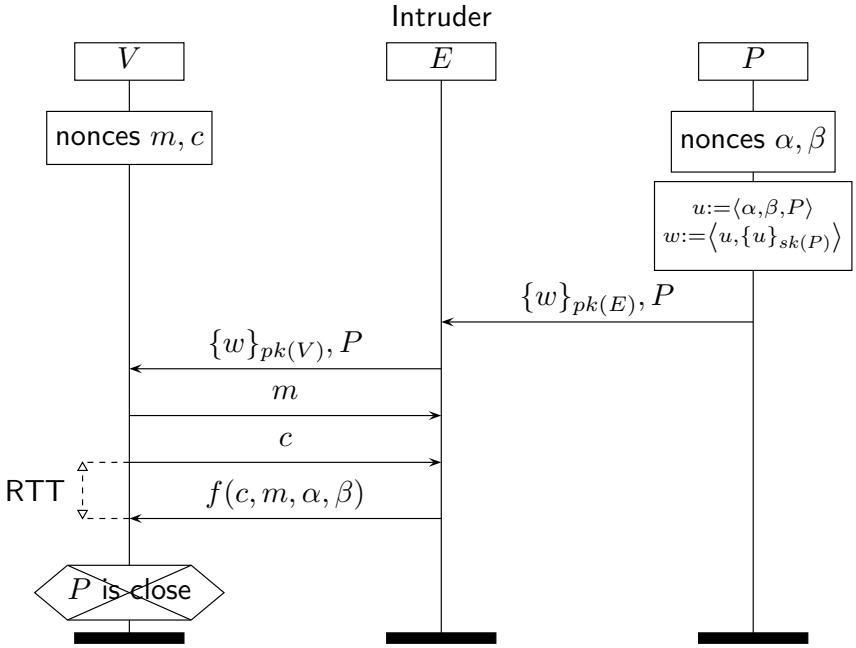


Figure 7.3 A mafia fraud attack on TREAD with asymmetric encryption.

V. Thus the intruder is prevented from re-encrypting the message received from P with the correct key. Nevertheless, a distance hijacking attack is possible, irrespective of the encryption scheme. The attack is represented in Figure 7.4.

Assume an honest prover P is close to the verifier V, whilst the intruder E is far from V. As before, P executes the protocol to prove its proximity to E. This allows E to learn α and β. Thus E starts a session with V by using the nonces α and β from P. At this point, V believes E is a legitimate prover and accept its signature. During the fast phase, P who is close to V, receives the challenge, supposedly from E, sent by V and replies correctly. Then V receives the response $f(c, m, \alpha, \beta)$, supposedly from E, sent by P who is close to V, and finishes the protocol with E correctly.

Neither of the two described attacks are possible when considering the adversary model used by the authors of the TREAD protocol. That is because their model does not allow for dishonest verifiers. In their model an honest prover will fail to initiate a communication with an untrusted verifier as the first message in each attack will not be sent. This adversary model is weaker than adversary models in the distance-bounding literature.

7.2 A Security Survey of Distance-Bounding Protocols

We conducted a security survey[2] of over 25 state-of-the-art distance-bounding protocols. For each of analyzed protocols, we verify whether it satisfies *dbsec_hnst* (without collusion), whether it satisfies *dbsec* (also without collusion) and whether it resists terrorist fraud (Definition 6.6). The results are shown in Table 7.1. In other chapter of this thesis (e.g. in Tables 4.1 and 4.2) the following acronyms have been used: BC for Brands and Chaum's protocol [BC93], RC for Rasmussen and Capkun's CRCS [RC10] protocol, Tree for Avoine

[2]Models and proofs available at https://github.com/jorgetp/dbverify

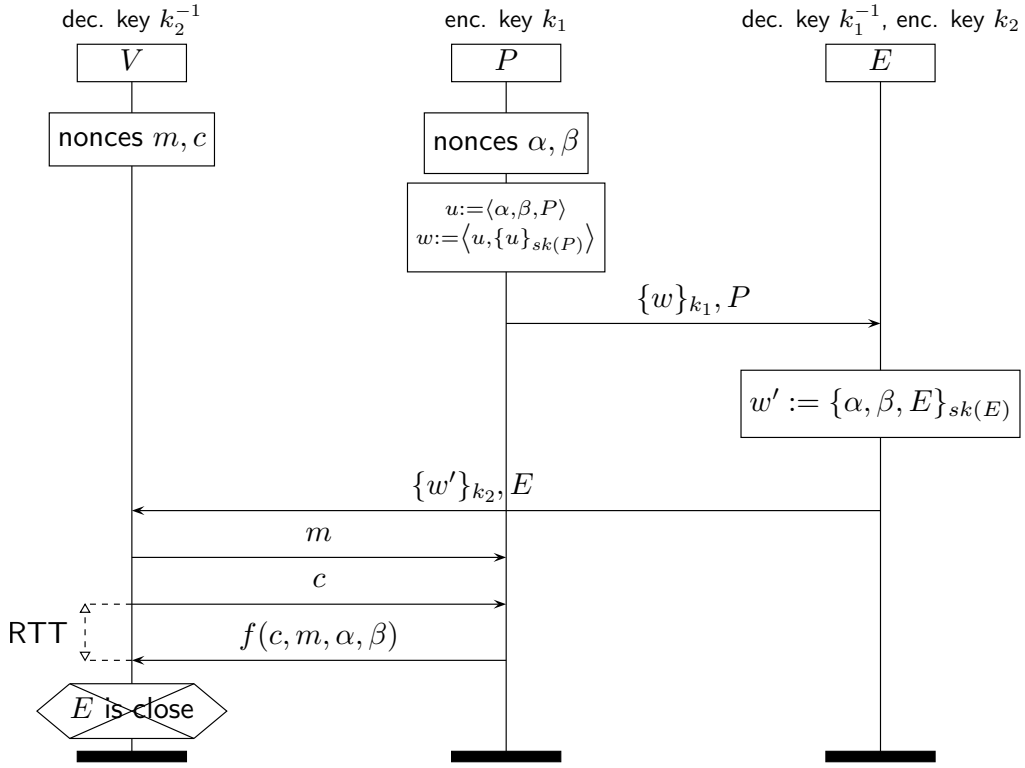


Figure 7.4 A distance hijacking attack on TREAD with symmetric encryption.

and Tchamkerten's protocol [AT09], HK for Hancke and Kuhn's protocol [HK05], MP for Munilla and Peinado's protocol [MP08], KA for Kim and Avoine's protocol [KA09], and BB for Bussard and Bagga's DBPK protocol [BB05].

To identify the type of attack (except for terrorist fraud) against a given protocol, if any, one may go through the outcome of Tamarin in interactive mode[3] and inspect the trace that invalidates the property *dbsec*. Two hints to identify such attacks are (1) if the protocol does not satisfy *dbsec_hnst*, then a mafia fraud exists; and (2) if the protocol satisfies *dbsec_hnst* but it does not satisfy *dbsec*, then a distance fraud exists, or a distance hijacking exists, or both.

Out of the analyzed protocols, only three protocols are distance-bounding secure and resist terrorist fraud. These protocols are Reid et al.'s [RNTS07], DBPK [BB05], and Swiss Knife [KAK+08]. A total of nineteen protocols were found vulnerable to terrorist fraud.

We note that the authors of UWB impulse radio based protocol [KLT10] do not give precise specifications of the *secure* channel used in the protocol. We took two different approaches when modeling this: a public key infrastructure and a keyed-MAC scheme. Both cases yield the same results, which include a mafia fraud against the protocol. Such attack is not reported in [KLT10] as the authors only consider verbatim relay.

For each one of the protocols reported as *not* resistant to terrorist fraud, the valid extension used to invalidate Equation 6.7 is the prover's leakage of the *least-disclosing* message. Such message is the knowledge-based minimal message that the adversary needs to produce the fast phase response, upon reception of the challenge. In most cases, if the

[3] See https://tamarin-prover.github.io/manual/index.html for instructions.

Table 7.1 TAMARIN verification results. The protocols that satisfy *dbsec* and resist terrorist fraud are highlighted in bold. The protocols from the block "Lookup-based" are a subset of protocols of the same name (studied in Part I) and have identical specification. Legend: ✓: verified, ×: falsified (i.e. attack found), $^{(n)}$: no symbolic (in)security proof reported before, and $^{(\neq c)}$: differs from Chothia et al.'s verification [CdRS18].

Protocol	**Satisfies** *dbsec_hnst*	**Satisfies** *dbsec*	**Resists terrorist fraud**
Brands and Chaum [BC93]			
- Signature id.	✓	×	$\times^{(n)}$
- Fiat-Shamir id.	✓	×	$\times^{(n)}$
CRCS [RC10]			
- Non-revealing sign.	✓	✓	×
- Revealing sign.	✓	×	×
Meadows et al. [MPP+07]			
- $f := \langle N_V, P \oplus N_P \rangle$	✓	$\times^{(\neq c)}$	×
- $f := N_V \oplus h(P, N_P)$	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
- $f := \langle N_V, P, N_P \rangle$	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
Lookup-based			
- Avoine and Tchamkerten [AT09]	✓	✓	$\times^{(n)}$
- Poulidor [TMA10]	✓	✓	$\times^{(n)}$
- Hancke and Kuhn [HK05]	✓	✓	$\times^{(\neq c)}$
- Uniform [MTT16a]	✓	✓	$\times^{(n)}$
Munilla and Peinado [MP08]	✓	✓	$\times^{(n)}$
Kim and Avoine [KA09]	✓	✓	$\times^{(n)}$
Reid et al. [RNTS07]	✓	✓	$\checkmark^{(n)}$
MAD (one way) [CBH03]	✓	$\times^{(\neq c)}$	×
DBPK [BB05]	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\checkmark^{(n)}$
Swiss Knife [KAK+08]	✓	✓	$\checkmark^{(n)}$
UWB [KLT10]			
- PKI	$\times^{(n)}$	$\times^{(n)}$	$\checkmark^{(n)}$
- keyed-MAC	$\times^{(n)}$	$\times^{(n)}$	$\checkmark^{(n)}$
WSBC+DB [PLHCT+10]	$\checkmark^{(n)}$	$\times^{(n)}$	$\times^{(n)}$
Hitomi [PCEvdL09]	$\checkmark^{(n)}$	$\checkmark^{(n)}$	$\times^{(n)}$
TREAD [ABG+17]			
- Asymmetric	×	×	$\checkmark^{(n)}$
- Symmetric	✓	×	$\checkmark^{(n)}$
ISO/IEC 14443			
- PaySafe [CGdR+15]	✓	×	×
- MIFARE Plus [TDJM+11]	✓	×	×
- PayPass [EMV18a]	✓	×	×

prover's fast phase response is of the form $f(ch, table)$ where ch is the verifier's fast phase challenge, then the least-disclosing message is $table$.

For each protocol *Proto* reported as resistant to terrorist fraud, one of the following three cases occurred:

(1) $Proto \not\models dbsec\_hnst$ and $Proto \not\models^{\star} dbsec\_hnst$, thus for any valid extension $Proto'$ of *Proto*, it holds that $Proto' \not\models^{\star} dbsec\_hnst$, given that $Tr(Proto) \subseteq Tr(Proto')$. The protocols within this case are TREAD [ABG+17] with asymmetric encryption, and both versions of UWB [KLT10].

(2) Every valid extension $Proto'$ of *Proto* such that $Proto' \not\models dbsec\_hnst$ leads to disclosure of the symmetric key shared by the prover and verifier, therefore $Proto' \not\models^{\star} dbsec\_hnst$. The protocols within this case are Reid et al. [RNTS07], DBPK [BB05], Swiss-Knife [KAK+08], and Hitomi [PCEvdL09].

(3) Every valid extension $Proto'$ of *Proto* such that $Proto' \not\models dbsec\_hnst$ leads to replay attacks, therefore $Proto' \not\models^{\star} dbsec\_hnst$. This is analogous to the analysis of *DBToy* in Theorem 6.1. The protocol within this case is TREAD [ABG+17] with symmetric encryption.

On average, a Tamarin model of a protocol consists of about 255 lines of code. On a modern laptop, the verification of all lemmas for a given protocol takes little over 1 minute on average and a few seconds in most cases. All (in)security proofs were constructed without any proof oracles for speeding up the verification.

Our Approach vs. Chothia et al.'s Approach

As briefly motivated in Chapter 2, Section 2.2, a recent publication [CdRS18] at the *USENIX Security Symposium 2018* analyzed a number of lookup-based protocols and reported that some of them are resistant to terrorist fraud. Our findings show incorrectness in their results.

Inconsistencies between our verification results and Chothia et al.'s are on the protocols Hancke and Kuhn's [HK05], Meadows et al.'s [MPP+07], and Capkun et al.'s MAD [CBH03]. On the one hand, Chothia et al.'s approach reports Hancke and Kuhn's protocol as resistant to terrorist fraud. It is well-known that this protocol does not resist this attack [BGL15, KAK+08, AMT15, ALM11, ABK+11, BMV13b, ABB+19]. On the other hand, Chothia et al.'s verification reports no attack, other than terrorist fraud, against Meadows et al.'s protocol version in which the prover's fast phase response is $\langle N_v, P \oplus N_p \rangle$, and the MAD protocol with one-way authentication. Our verification, however, identifies a valid distance hijacking attack against each of these protocols. Indeed, a number of previous works (e.g. [CRSC12, AMT15, MSTT18b, DDW18]) report such attacks as well.

Finally, we observe that Chothia et al.'s theoretical model seems to be attached to the ProVerif prover, possibly due to the way that co-location of processes is represented. However, ProVerif is not particularly suited to deal with protocols that feature stateful data such as session counters, which are used in numerous protocols. Thus, ProVerif-based frameworks such as Chothia et al.'s, oversimplify such protocols by replacing those

features with simpler constructions, which might lead to missing attacks. A similar over-approximation issue was pointed out, although in the context of authentication protocols, in [BDH+18].

Tamarin comes with a built-in `multiset` which can be used to faithfully model counters, thus making the verification more accurate. In addition, our theoretical model is not exclusive to Tamarin. Indeed, it can be also implemented in Isabelle/HOL [NPW02] as we build upon Mauw et al.'s model [MSTT18b], which is in turn a refinement of the Isabelle/HOL-supported model by Basin et al. [BCSS09, SSBC09].

7.3 On the ISO/IEC 14443 Protocols

The ISO/IEC 14443 standard is used in more than 80% of today's contactless smart cards. Within our case studies, we analyzed three protocols based on this standard:

- NXP's MIFARE Plus[4] (versions X and EV1) with proximity check (with patent [TDJM+11]) with worldwide applications in public transport, access management, school and campus cards, citizen cards, employee cards, and car parking.
- PaySafe [CGdR+15], which is a distance-bounding-enabled version of Visa's contactless payment protocol payWave (in qVSDC mode) [EMV18b].
- PayPass [EMV18a], which is Mastercard's contactless payment protocol with relay resistance.

To demonstrate our analysis, we have chosen the PayPass protocol, represented in Figure 7.5. The analyses of the other 2 protocols are analogous. In the context of these protocols, the verifier R is the reader terminal and the prover C is the card. The PayPass protocol is a relay-resistance-enabled version of the EMV[5] payment protocol implemented in Mastercard's contactless cards. EMV (which stands for Europay, Mastercard and Visa) has become the international standard for smart cards/chips payment protocols.

In a regular EMV session, a transaction is initiated by the exchange of SELECT and SELECTED commands along with the selected EMV applet that will be used for the transaction (PayPass in this case). Then, the terminal issues the GPO command to inform the card on the terminal's capabilities. The card then responds to this command with the Application Interchange Profile (AIP) and Application File Locator (AFL) which indicate the card's capabilities and the location of data files, respectively. Then, the terminal issues the GENERATE_AC command, which includes an Unpredictable Number UN, the amount of the transaction, the currency code, and other data. The cards responds with the Application Cryptogram (AC), the Signed Dynamic Application Data ($SDAD$) and the Application Transaction Counter (ATC). The AC is a the result of keyed-MAC on the transaction information whose key is an encryption of the Application Transaction Counter (ATC, equal to the number of transactions previously made by the card) with a long-term symmetric key shared between the terminal and the card. The AC is the proof of the transaction, which can be verified by the card issuer. The $SDAD$ is the card's signature on the transaction.

[4] https://www.mifare.net/en/products/chip-card-ics/mifare-plus/
[5] https://www.emvco.com

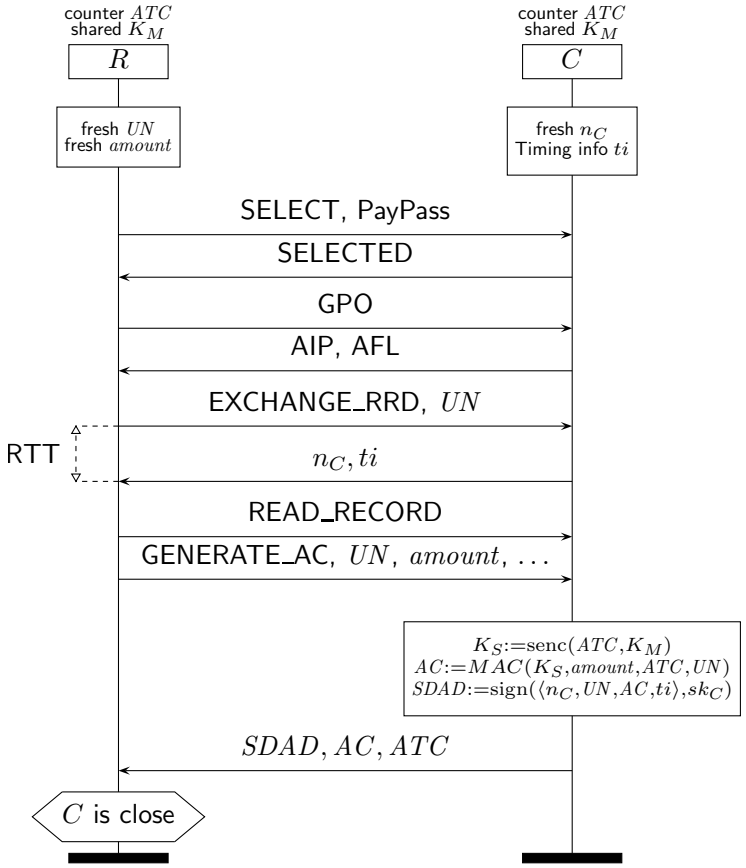


Figure 7.5 Mastercard's PayPass protocol.

To enable the EMV protocol with the relay resistance mechanism, after the card issues the AIP and AFL commands, the terminal issues the new Exchange Relay Resistance Data EXCHANGE_RRD command, along with the Terminal Relay Resistance Entropy number (which equals UN). This message initiates the fast phase of the protocol. The card must respond on time with their nonce n_C (Device Relay Resistance Entropy) and three timing estimates (minimum time for processing, maximum time for processing and estimated transmission time). The maximum time serves as an upper bound for the terminal's timer.

When modeling the PayPass protocol in Tamarin, and also the other ISO/IEC 14443 protocols, we made the following abstractions: (1) the timing information is considered a nonce; and (2) we did not model any exchanged messages that are fully composed of constant terms, e.g. the first message ⟨SELECT, PayPass⟩.

As Table 7.1 shows, PayPass satisfies *dbsec_hnst*, which means that it does resist mafia fraud and in particular, relay attacks. Indeed, defending against relay is a fundamental security goal of this protocol. However, PayPass fails to defend against distance fraud [Des88] and distance hijacking [CRSC12]. Those attacks refer to a remote and compromised card which successfully tricks the reader into believing they are co-located, and thus the reader accepts the transaction.

One might argue that those attacks are irrelevant for payment systems. After all, it is the compromised card's owner's bank account which ends up being charged. However, suppose an attacker has acquired the payment card of a victim and wishes to cause them

harm. After compromising the card, they might place a concealed device near the checkout area of a store that performs a distance hijacking attack using the compromised card. Shoppers at the store would then perform transactions, believing that they were paying for products, whilst in fact all payments came from the one corrupted card. The attacker could even mix in several transactions of their own, which would be indistinguishable from the honest shoppers. As a result of this "Robin Hood" style attack, the victim will be charged for these illegitimate transactions with no clear perpetrator.

Fixing the ISO/IEC 14443 Protocols

As before, we will focus on the PayPass protocol. Before giving the fixes, let us motivate the reasons for which it does not satisfy *dbsec*. As noted by Mauw et al. in [MSTT18b] when analyzing the PaySafe protocol, a distance fraud attack is possible due to the lack of a causal relation between the fast phase challenge and response. That is, the fast phase response can be produced prior to reception of the challenge. To solve this issue, Mauw et al. suggested the inclusion of the reader's nonce UN within the card's response.

Mauw et al.'s suggestion applied on PayPass does prevent distance fraud, but it does not prevent distance hijacking. To prevent the latter, we must bind the fast phase messages to the card's identity. We do so by adding to the card's fast phase response, besides UN, the card's signature on the nonce n_C. Thus, the card's fast phase response becomes:

$$\langle n_C, ti, \mathrm{sign}(n_C, sk_C), UN \rangle\,.$$

This modification results in a protocol, represented in Figure 7.6, that satisfies *dbsec*. Observe that the signature $\mathrm{sign}(n_C, sk_C)$ can be computed prior to the fast phase, so it does not delay the card's response. The very same solution of adding $\langle \mathrm{sign}(n_C, sk_C), UN \rangle$ into the card's fast phase response also works for both the PaySafe and MIFARE Plus protocols. Though, to keep consistency with the usage of cryptographic operations in the case of the latter protocol, we propose a keyed-MAC message $MAC(K_M, n_C, \text{'1'}, \text{'2'})$ instead of the signature $\mathrm{sign}(n_C, sk_C)$. As before, the keyed-MAC message can be computed prior to the fast phase as well.

The proposed version (Figure 7.6) of the PayPass protocol does *not* resist terrorist fraud. Indeed, the card's leakage of $\langle n_C, ti, \mathrm{sign}(n_C, sk_C) \rangle$ prior to the fast phase leads to a valid attack. To prevent terrorist fraud, we propose to further modify the PayPass protocol by changing the card's fast phase response and $SDAD$ messages so that they become:

$$\langle n_C, ti, f(UN, n_C \oplus K_M) \rangle \quad \text{and} \quad \mathrm{sign}(\langle UN, AC \rangle, sk_C),$$

respectively; where f is an irreversible function. This modification results in a protocol, shown in Figure 7.7, that satisfies *dbsec* and resists terrorist fraud.

Similar constructions can be done over the PaySafe and MIFARE Plus protocols to repair them. Though only the modification of the card's fast phase response is required to repair the latter protocol. The Tamarin models and security proofs of the two versions of each protocol are available in our repository. We give two different repaired versions of each protocol in order to leave the choice up to the requirements of the application system. For example, if terrorist fraud is not a critical issue, then the first modification is suggested over the second one, because the latter modifies the standard more "aggressively". We do always suggest the first modified version over the original protocol, regardless of the application.

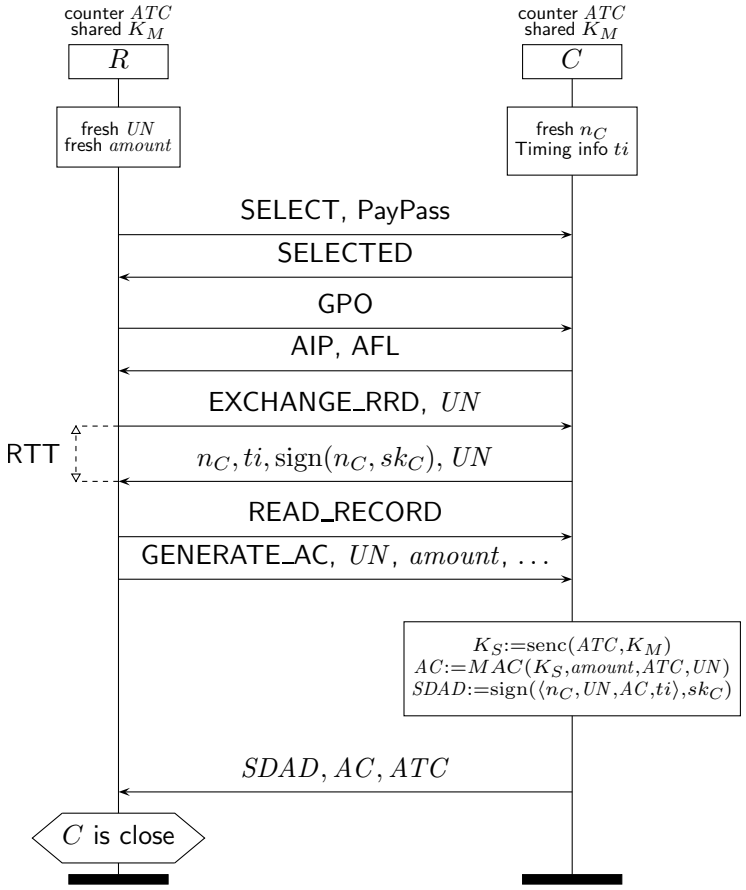


Figure 7.6 A modified version of Mastercard's PayPass protocol that satisfies *dbsec*. The modification to the original protocol (Figure 7.5) consists of the addition of $\text{sign}(n_C, sk_C)$ and UN to the card's fast phase response.

Other modifications for ISO/IEC 14443 protocols that make them resistant to terrorist fraud possibly exist, and likely all of them (like ours) would require major changes to the standard. For example, the composition of the *SDAD* message would likely have to be modified due to the occurrence of the card's nonces within it. Furthermore, we conjecture that if the card's nonces (e.g. n_C) can be inferred from passive observation of the execution, then versions of the protocols in question that resist terrorist fraud would be vulnerable to relay attacks, thus violating a primary security goal of these protocols.

7.4 Conclusions

We provided computer-verifiable proofs of the (in)security of over 25 distance-bounding protocols that account for all classes of attacks, as given by the literature on distance bounding. To the best of our knowledge, this is the most extensive and sound set of security/vulnerability proofs within this research subject. Our verification reports that for the vast majority of the analyzed protocols at least one attack exists. The vulnerable protocols include protocols based on the ISO/IEC 14443 standard such as Mastercard's PayPass [EMV18a], a distance-bounding-enabled version [CGdR+15] of Visa's

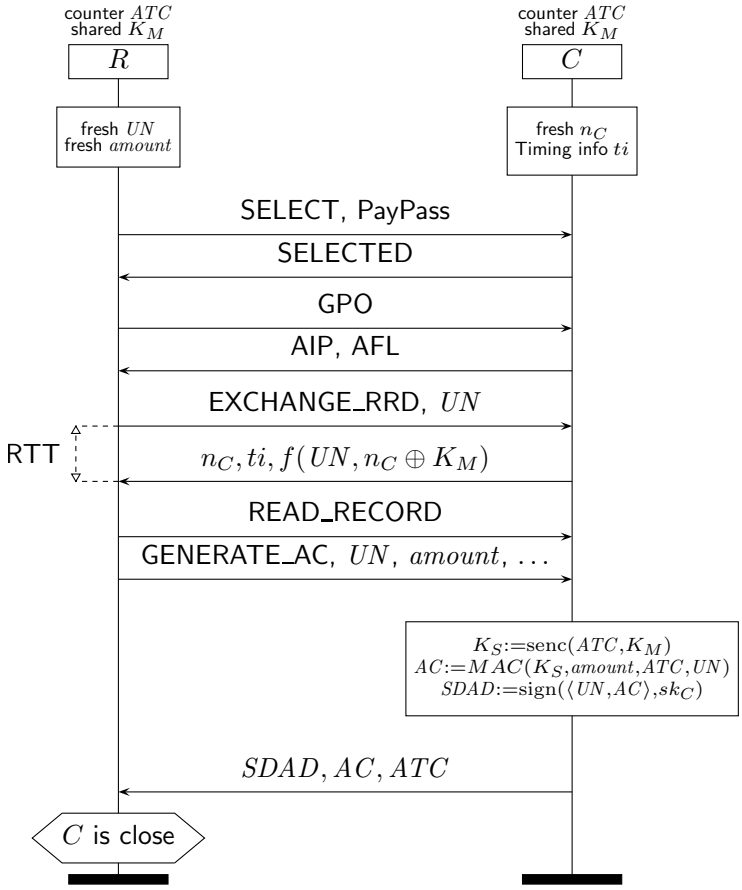


Figure 7.7 A modified version of Mastercard's PayPass protocol that satisfies *dbsec* and resists terrorist fraud. The modifications to the original protocol (Figure 7.5) are: (1) addition of $f(UN, n_C \oplus K_M)$ to the card's fast phase response, and (2) removal of n_C and ti from the $SDAD$ message.

payWave [EMV18b], and NXP's MIFARE Plus with proximity check [TDJM+11]. Finally, we proposed fixes for these protocols and provide computer-verifiable security proofs of the repaired protocols. The proposed fixes form demonstrative examples that could be used to improve proximity-based secure systems that follow the standard, or may even form guidance for a future version of the standard itself.

8

Conclusions

In this thesis we have addressed the topic of security analysis of distance-bounding protocols. The fundamental goal of this thesis is to provide researchers and industry professionals with frameworks that allow them to analyze the security of these protocols in a systematic manner. The proposed frameworks, like traditional frameworks for security protocol verification, conceive the distance-bounding security models in the two classic settings: computational and symbolic. Computational security proofs are assessed against arbitrary probabilistic polynomial-time algorithms, thus offering strong security guarantees. Symbolic security proofs, on the other hand, follow from a high-level, logic analysis of the protocol executions, and treat cryptographic primitives as unbreakable black-boxes.

On Computational Analysis

We developed an automata-based model for systematic analysis of a large class of distance-bounding protocols, called lookup-based protocols. The protocols composing this class are those in which (1) the prover's responses to the verifier's fast phase challenges are the result of a lookup operation from a table built up in the initial phase, and (2) the prover does not send any messages after the fast phase. Our proposal represents a protocol as a finite set of State-Labeled Deterministic Finite Automata, and a protocol execution is thus represented by a random walk in a randomly selected automaton. The adversary is modeled by a polynomial-time algorithm whose input is the walk in the selected automaton.

By using the proposed automata-based model, we demonstrated that no lookup-based protocol has better provable-resistance to mafia fraud attacks than Avoine and Tchamkerten's Tree protocol [AT09]. We also defined an optimal pre-ask strategy for an adversary to perform a mafia fraud attack against any lookup-based protocol that is layered (different-length sequences do not reach the same state) and random-labeled (any modification to the labeling function of an automaton of the protocol results in a automaton that is also in the protocol). The strategy consists in replying to the verifier's challenges with exactly the same responses the adversary obtained from the prover in the pre-ask session. This strategy allows us to straightforwardly compute probability of success of a mafia fraud attack (based on the pre-ask strategy) against most lookup-based protocols proposed to date.

Motivated by the exponential space complexity of the optimally secure Tree protocol, we propose a family of protocols that strike excellent resistance to pre-ask attacks in relation to their space complexity. Every protocol of this family satisfies that, a positive integer

number u exists, such that two input sequences meet, for each automaton, if and only if they have the same u-length suffix.

We defined structural equivalence relations between automata that allow us to take the security and space complexity analysis even further. Indeed, we have proposed the first lookup-based distance-bounding protocol that is optimal in terms of mafia fraud resistance, amongst all layered and random-labeled protocols with an upper bound on the size of the protocol. The size of a layered protocol is the number of states of the largest layer amongst all automata of the protocol. The size of a protocol defines its space complexity, which in turns defines the amount of memory the protocol requires. The proposed optimal protocol, called the Modular protocol, is composed of automata whose transition functions are based on modular congruences. As a result, its automata have the largest girth (the girth is the shortest cycle if viewed as an undirected graph) amongst all layered protocol with equal or smaller size. We also provided a concrete cryptographic construction of the Modular protocol.

We conducted a comprehensive set of experiments supported by the decision-making theory [AMT15], which included thirteen state-of-the-art protocols. The experiments indicate that the proposed protocol cannot be outperformed by other protocols.The Modular protocol seems to have interesting connections with previous work in other security fields. For example, the properties of bipartite and q-partite graphs, which relates to layered protocols, have been used to guarantee lower bounds on the code rate of error correcting codes [Tan81, KPP$^+$04]. Moreover, graphs with large girth, such as expander and Cayley graphs, have been studied for years in order to design provable-secure cryptographic hashes [Zém91].

The downside of framework approach is that we do not provide methods for generic analysis of attacks other than mafia fraud. Our belief though is that our automata-based representation does allow one to systematically reason on security against further attacks.

On Symbolic Analysis

We studied the symbolic verification framework proposed by Basin et al. [BCSS09, SSBC09]. This framework is based on timed-events and agents' locations and it delivers Isabelle/HOL-constructed (in)security proofs. With focus on this work, we characterize a semantic domain of well-formed distance-bounding protocols in which the timestamps associated to the agents' actions are only utilized for proximity verification purposes and not for, e.g., taking a different branch in the execution. This is not a trivial class, but it contains most (if not all) distance-bounding protocols published to date. We proposed the first causality-based security property for symbolic verification of distance-bounding protocols which we prove equivalent to Basin et al.'s property, for any well-formed distance-bounding protocol.

Our property resembles the aliveness authentication property [Low97, CM12] in that the prover (or any compromised agent if such prover is compromised) must perform some action during the fast phase. We demonstrated that, for any well-formed distance-bounding protocol, our property is equivalent to the statement that the fast phase round-trip time is an upper bound on twice the prover-to-verifier distance multiplied by the network speed.

We refined the traditional Dolev-Yao [DY83] adversary model to account for collusion in security protocols in the presence of colluding agents. Colluding agents are protocol participants who are not under full control of the adversary, yet they are willing to deviate

from the intended protocol specification with the goal of breaking a given security property. We introduced the notion of post-collusion security, which provides security guarantees even if the agents involved have been colluding in previous sessions of the protocol. By means of post-collusion security, we provided a symbolic definition of (resistance to) terrorist fraud. This is a non-standard class of attack in which a distant non-compromised prover colludes colludes with a compromised prover who is co-located with the verifier in order to make the verifier believe that the distant prover is proximal. As condition on the collusion is that the same proximity proof cannot be provided in further sessions without further collusion.

We built a Tamarin-based verification framework that accounts for all classes of attacks, as given by the literature on distance bounding. With our framework, we conducted verification of a over 25 state-of-the-art distance-bounding protocols. To the best of our knowledge, this is the most extensive and sound set of security/vulnerability proofs within this research subject. Our verification reports that for the vast majority of the analyzed protocols at least one attack exists. The vulnerable protocols include industrial protocols based on the ISO/IEC 14443 standard such as Mastercard's PayPass [EMV18a], a distance-bounding-enabled version [CGdR+15] of Visa's payWave [EMV18b], and NXP's MIFARE Plus with proximity check [TDJM+11]. We provided fixes for these protocols as well as computer-verifiable security proofs of the repaired protocols. The proposed fixes form demonstrative examples that could be used to improve proximity-based secure systems that follow the standard, or may even form guidance for future versions of the standard itself.

Unfortunately, the Tamarin prover together with our definition of resistance to terrorist fraud do not make a *complete* verification approach. That is, there may exist a protocol that resists terrorist fraud (as per our definition) but Tamarin is unable to deliver proof of it. The cause goes beyond the well-acknowledged non-termination issue of the tool, as it is due to the impossibility of covering up the full set of possible collusion actions.

Future Work

As future work, we will look into the mentioned issues: (1) to broaden the focus of the computational analysis of security to attacks other than mafia fraud, and (2) the completeness issue of our theoretical definition of terrorist fraud resistance in relation to its applicability in Tamarin.

Furthermore, it would be beneficial to reduce the gap between computational and symbolic models. This seems to be a research direction that one would benefit from, not only for analysis of distance-bounding protocols, but for security protocols in general. Specifically, we would like to develop an (ideally) automated framework that employs the techniques from both the computational and the symbolic approaches.

A possible direction would be the integration of more realistic equational theories into multiset rewriting models that would make cryptographic primitives "less symbolic". One can think of, for example, an equational theory to account for weak hashes or weak random generators. Another idea is to attach a term to each message that represents the probability that the adversary knows such a message. Hence, modifier rules for such probabilities could be worked around, e.g. if a message is sent, then the attached probability term to such message becomes *one*. Therefore, the adversary's inference of messages can be modeled in a probabilistic fashion. Obviously these suggestions imply that the overall verification time

will increase considerably, therefore optimization techniques will be needed on both the tool and the user sides.

Bibliography

[ABB+19] Gildas Avoine, Muhammed Ali Bingöl, Ioana Boureanu, Srdjan Capkun, Gerhard P. Hancke, Süleyman Kardas, Chong Hee Kim, Cédric Lauradoux, Benjamin Martin, Jorge Munilla, Alberto Peinado, Kasper Bonne Rasmussen, Dave Singelée, Aslan Tchamkerten, Rolando Trujillo-Rasua, and Serge Vaudenay. Security of distance-bounding: A survey. *ACM Comput. Surv.*, 51(5):94:1–94:33, 2019.

[ABG+17] Gildas Avoine, Xavier Bultel, Sébastien Gambs, David Gérault, Pascal Lafourcade, Cristina Onete, and Jean-Marc Robert. A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pages 800–814, 2017.

[ABK+11] Gildas Avoine, Muhammed Ali Bingöl, Süleyman Kardas, Cédric Lauradoux, and Benjamin Martin. A framework for analyzing RFID distance bounding protocols. *Journal of Computer Security*, 19(2):289–317, 2011.

[AL10] Yonatan Aumann and Yehuda Lindell. Security against covert adversaries: Efficient protocols for realistic adversaries. *J. Cryptology*, 23(2):281–343, 2010.

[ALM11] Gildas Avoine, Cédric Lauradoux, and Benjamin Martin. How secret-sharing can defeat terrorist fraud. In *Proceedings of the Fourth ACM Conference on Wireless Network Security, WISEC 2011, Hamburg, Germany, June 14-17, 2011*, pages 145–156, 2011.

[AMT15] Gildas Avoine, Sjouke Mauw, and Rolando Trujillo-Rasua. Comparing distance bounding protocols: A critical mission supported by decision theory. *Computer Communications*, 67:92–102, 2015.

[AT09] Gildas Avoine and Aslan Tchamkerten. An efficient distance bounding RFID authentication protocol: Balancing false-acceptance rate and memory requirement. In *Information Security, 12th International Conference, ISC 2009, Pisa, Italy, September 7-9, 2009. Proceedings*, pages 250–261, 2009.

[BB05] Laurent Bussard and Walid Bagga. Distance-bounding proof of knowledge to avoid real-time attacks. In *Security and Privacy in the Age of Ubiquitous*

Computing, IFIP TC11 20th International Conference on Information Security (SEC 2005), May 30 - June 1, 2005, Chiba, Japan, pages 223–238, 2005.

[BC93] Stefan Brands and David Chaum. Distance-bounding protocols (extended abstract). In *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, pages 344–359, 1993.

[BCSS09] David A. Basin, Srdjan Capkun, Patrick Schaller, and Benedikt Schmidt. Let's get physical: Models and methods for real-world security protocols. In *Theorem Proving in Higher Order Logics, 22nd International Conference, TPHOLs 2009, Munich, Germany, August 17-20, 2009. Proceedings*, pages 1–22, 2009.

[BD90] Thomas Beth and Yvo Desmedt. Identification tokens - or: Solving the chess grandmaster problem. In *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, pages 169–177, 1990.

[BDH+18] David A. Basin, Jannik Dreier, Lucca Hirschi, Sasa Radomirovic, Ralf Sasse, and Vincent Stettler. A formal analysis of 5G authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 1383–1396, 2018.

[BGG+16] Xavier Bultel, Sébastien Gambs, David Gerault, Pascal Lafourcade, Cristina Onete, and Jean-Marc Robert. A prover-anonymous and terrorist-fraud resistant distance-bounding protocol. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, WISEC 2016, Darmstadt, Germany, July 18-22, 2016*, pages 121–133, 2016.

[BGL15] Agnès Brelurut, David Gerault, and Pascal Lafourcade. Survey of distance bounding protocols and threats. In *Foundations and Practice of Security - 8th International Symposium, FPS 2015, Clermont-Ferrand, France, October 26-28, 2015, Revised Selected Papers*, pages 29–49, 2015.

[Bla00] Matt Blaze. Looking on the bright side of black-box cryptography (transcript of discussion). In *Security Protocols, 8th International Workshop, Cambridge, UK, April 3-5, 2000, Revised Papers*, pages 54–61, 2000.

[Bla01] Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14 2001), 11-13 June 2001, Cape Breton, Nova Scotia, Canada*, pages 82–96, 2001.

[BMV13a] Ioana Boureanu, Aikaterini Mitrokotsa, and Serge Vaudenay. Secure and lightweight distance-bounding. In *Proc. 2nd International Workshop on Lightweight Cryptography for Security and Privacy (LightSec'13)*, volume 8162 of *LNCS*, pages 97–113, Gebze, Turkey, May 2013. Springer-Verlag.

[BMV13b] Ioana Boureanu, Aikaterini Mitrokotsa, and Serge Vaudenay. Towards secure distance bounding. In *Fast Software Encryption - 20th International Workshop, FSE 2013, Singapore, March 11-13, 2013. Revised Selected Papers*, pages 55–67, 2013.

[BV14] Ioana Boureanu and Serge Vaudenay. Optimal proximity proofs. In *Information Security and Cryptology - 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers*, pages 170–190, 2014.

[CBH03] Srdjan Capkun, Levente Buttyán, and Jean-Pierre Hubaux. SECTOR: secure tracking of node encounters in multi-hop wireless networks. In *Proceedings of the 1st ACM Workshop on Security of ad hoc and Sensor Networks, SASN 2003, Fairfax, Virginia, USA, 2003*, pages 21–32, 2003.

[CC03] Hubert Comon-Lundh and Véronique Cortier. Security properties: Two agents are sufficient. In *Programming Languages and Systems, 12th European Symposium on Programming, ESOP 2003, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7-11, 2003, Proceedings*, pages 99–113, 2003.

[CdRS18] Tom Chothia, Joeri de Ruiter, and Ben Smyth. Modelling and analysis of a hierarchy of distance bounding attacks. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.*, pages 1563–1580, 2018.

[CEJvO02] Stanley Chow, Philip A. Eisen, Harold Johnson, and Paul C. van Oorschot. White-box cryptography and an AES implementation. In *Selected Areas in Cryptography, 9th Annual International Workshop, SAC 2002, St. John's, Newfoundland, Canada, August 15-16, 2002. Revised Papers*, pages 250–270, 2002.

[CGdR+15] Tom Chothia, Flavio D. Garcia, Joeri de Ruiter, Jordi van den Breekel, and Matthew Thompson. Relay cost bounding for contactless EMV payments. In *Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers*, pages 189–206, 2015.

[CHH+17] Cas Cremers, Marko Horvat, Jonathan Hoyland, Sam Scott, and Thyla van der Merwe. A comprehensive symbolic analysis of TLS 1.3. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, pages 1773–1788, 2017.

[CHKM06] Jolyon Clulow, Gerhard P. Hancke, Markus G. Kuhn, and Tyler Moore. So near and yet so far: Distance-bounding attacks in wireless networks. In *Security and Privacy in Ad-Hoc and Sensor Networks, Third European Workshop, ESAS 2006, Hamburg, Germany, September 20-21, 2006, Revised Selected Papers*, pages 83–97, 2006.

[CKW11] Véronique Cortier, Steve Kremer, and Bogdan Warinschi. A survey of symbolic methods in computational analysis of cryptographic systems. *J. Autom. Reasoning*, 46(3-4):225–259, 2011.

[CM12] Cas Cremers and Sjouke Mauw. *Operational Semantics and Verification of Security Protocols*. Springer, 2012.

[CMP05] Iliano Cervesato, Catherine A. Meadows, and Dusko Pavlovic. An encapsulated authentication logic for reasoning about key distribution protocols. In *18th IEEE Computer Security Foundations Workshop, (CSFW-18 2005), 20-22 June 2005, Aix-en-Provence, France*, pages 48–61, 2005.

[CO99] Ran Canetti and Rafail Ostrovsky. Secure computation with honest-looking parties: What if nobody is truly honest? (extended abstract). In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1-4, 1999, Atlanta, Georgia, USA*, pages 255–264, 1999.

[Con76] John Horton Conway. *On Numbers and Games*. Academic Press Inc., London, UK, 1976.

[Cre08] Cas J. F. Cremers. The Scyther tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference, CAV 2008, Princeton, NJ, USA, July 7-14, 2008, Proceedings*, pages 414–418, 2008.

[CRSC12] Cas J. F. Cremers, Kasper Bonne Rasmussen, Benedikt Schmidt, and Srdjan Capkun. Distance hijacking attacks on distance bounding protocols. In *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*, pages 113–127, 2012.

[DDW18] Alexandre Debant, Stéphanie Delaune, and Cyrille Wiedling. A symbolic framework to analyse physical proximity in security protocols. In *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, pages 29:1–29:20, 2018.

[Des88] Yvo Desmedt. Major security problems with the 'unforgeable' (Feige-)Fiat-Shamir proofs of identity and how to overcome them. In *SECURICOM'88*, pages 15–17, 1988.

[DFKO11] Ulrich Dürholz, Marc Fischlin, Michael Kasper, and Cristina Onete. A formal approach to distance-bounding RFID protocols. In *Information Security, 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings*, pages 47–62, 2011.

[DGB88] Yvo Desmedt, Claude Goutier, and Samy Bengio. Special uses and abuses of the Fiat-Shamir passport protocol. In *Proc. Advances in Cryptology (CRYPTO'87)*, volume 293 of *LNCS*, pages 21–39. Springer, 1988.

[DY83] Danny Dolev and Andrew Chi-Chih Yao. On the security of public key protocols. *IEEE Trans. Information Theory*, 29(2):198–207, 1983.

[EMV18a] EMVCo. *EMV Contactless Specifications for Payment Systems, Book C-2, Kernel 2 Specification, Version 2.7.* April 2018.

[EMV18b] EMVCo. *EMV Contactless Specifications for Payment Systems, Book C-3, Kernel 3 Specification, Version 2.7.* April 2018.

[FDC11] Aurélien Francillon, Boris Danev, and Srdjan Capkun. Relay attacks on passive keyless entry and start systems in modern cars. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*, 2011.

[FY92] Matthew K. Franklin and Moti Yung. Communication complexity of secure computation (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 699–710, 1992.

[GAA10] Ali Özhan Gürel, Atakan Arslan, and Mete Akgün. Non-uniform stepping approach to RFID distance bounding problem. In *Data Privacy Management and Autonomous Spontaneous Security - 5th International Workshop, DPM 2010 and 3rd International Workshop, SETOP 2010, Athens, Greece, September 23, 2010, Revised Selected Papers*, pages 64–78, 2010.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[HK05] Gerhard P. Hancke and Markus G. Kuhn. An RFID distance bounding protocol. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks, SecureComm 2005, Athens, Greece, 5-9 September, 2005*, pages 67–73, 2005.

[HK08] Gerhard P. Hancke and Markus G. Kuhn. Attacks on time-of-flight distance bounding channels. In *Proceedings of the First ACM Conference on Wireless Network Security, WISEC 2008, Alexandria, VA, USA, March 31 - April 02, 2008*, pages 194–202, 2008.

[KA09] Chong Hee Kim and Gildas Avoine. RFID distance bounding protocol with mixed challenges to prevent relay attacks. In *Proc. 8th International Conference on Cryptology and Network Security (CANS'09)*, volume 5888 of *LNCS*, pages 119–133. Springer, 2009.

[KA11] Chong Hee Kim and Gildas Avoine. RFID distance bounding protocols with mixed challenges. *IEEE Transactions on Wireless Communications*, 10(5):1618–1626, 2011.

[KAK+08] Chong Hee Kim, Gildas Avoine, François Koeune, François-Xavier Standaert, and Olivier Pereira. The Swiss-Knife RFID distance bounding protocol. In *Information Security and Cryptology - ICISC 2008, 11th International Conference, Seoul, Korea, December 3-5, 2008, Revised Selected Papers*, pages 98–115, 2008.

[KKBD11] Süleyman Kardas, Mehmet Sabir Kiraz, Muhammed Ali Bingöl, and Hüseyin Demirci. A novel RFID distance bounding protocol based on physically unclonable functions. In *RFID. Security and Privacy - 7th International Workshop, RFIDSec 2011, Amherst, USA, June 26-28, 2011, Revised Selected Papers*, pages 78–93, 2011.

[KLT10] Marc Kuhn, Heinrich Luecken, and Nils Ole Tippenhauer. UWB impulse radio based distance bounding. In *7th Workshop on Positioning Navigation and Communication, WPNC 2010, Dresden Germany, 11-12 March 2010, Proceedings*, pages 28–37, 2010.

[KPP+04] Jon-Lark Kim, Uri N. Peled, I. Perepelitsa, Vera Pless, and Shmuel Friedland. Explicit construction of families of LDPC codes with no 4-cycles. *IEEE Trans. Information Theory*, 50(10):2378–2388, 2004.

[Low97] Gavin Lowe. A hierarchy of authentication specification. In *10th Computer Security Foundations Workshop (CSFW '97), June 10-12, 1997, Rockport, Massachusetts, USA*, pages 31–44, 1997.

[MBK10] Sreekanth Malladi, Bezawada Bruhadeshwar, and Kishore Kothapalli. Automatic analysis of distance bounding protocols. *CoRR*, abs/1003.5383, 2010.

[MP08] Jorge Munilla and Alberto Peinado. Distance bounding protocols for RFID enhanced by using void-challenges and analysis in noisy channels. *Wireless Communications and Mobile Computing*, 8(9):1227–1232, 2008.

[MPP+07] Catherine A. Meadows, Radha Poovendran, Dusko Pavlovic, LiWu Chang, and Paul F. Syverson. Distance bounding protocols: Authentication logic analysis and collusion attacks. In *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*, pages 279–298. 2007.

[MSCB13] Simon Meier, Benedikt Schmidt, Cas Cremers, and David A. Basin. The TAMARIN prover for the symbolic analysis of security protocols. In *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, pages 696–701, 2013.

[MSTT18a] Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Automated identification of desynchronisation attacks on shared secrets. In *Computer Security - 23rd European Symposium on Research in, ESORICS 2018, Barcelona, Spain, September 3-7, 2018, Proceedings, Part I*, pages 406–426, 2018.

[MSTT18b] Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *2018 IEEE Symposium on Security and Privacy, S&P 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 549–566, 2018.

[MSTT19] Sjouke Mauw, Zach Smith, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Post-collusion security and terrorist fraud. 2019. (under submission).

[MTT16a] Sjouke Mauw, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. A class of precomputation-based distance-bounding protocols. In *IEEE European Symposium on Security and Privacy, EuroS&P 2016, Saarbrücken, Germany, March 21-24, 2016*, pages 97–111, 2016.

[MTT16b] Sjouke Mauw, Jorge Toro-Pozo, and Rolando Trujillo-Rasua. Optimality results on the security of lookup-based protocols. In *Radio Frequency Identification and IoT Security - 12th International Workshop, RFIDSec 2016, Hong Kong, China, November 30 - December 2, 2016, Revised Selected Papers*, pages 137–150, 2016.

[NPW02] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel. *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, 2002.

[NS87] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Commun. ACM*, 21(12), 1987.

[PCEvdL09] Pedro Peris-Lopez, Julio César Hernández Castro, Juan M. Estévez-Tapiador, and Jan C. A. van der Lubbe. Shedding some light on RFID distance bounding protocols and terrorist attacks. *CoRR*, abs/0906.4618, 2009.

[PLHCT+10] P. Peris-Lopez, J. C. Hernandez-Castro, J. M. E. Tapiador, E. Palomar, and J. C. A. van der Lubbe. Cryptographic puzzles and distance-bounding protocols: Practical tools for rfid security. In *2010 IEEE International Conference on RFID (IEEE RFID 2010)*, pages 45–52, 2010.

[PM06] Dusko Pavlovic and Catherine A. Meadows. Deriving secrecy in key establishment protocols. In *Computer Security - ESORICS 2006, 11th European Symposium on Research in Computer Security, Hamburg, Germany, September 18-20, 2006, Proceedings*, pages 384–403, 2006.

[RC10] Kasper Bonne Rasmussen and Srdjan Capkun. Realization of RF distance bounding. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 389–402, 2010.

[RNTS07] Jason Reid, Juan Manuel González Nieto, Tee Tang, and Bouchra Senadji. Detecting relay attacks with timing-based protocols. In *Proceedings of the 2007 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2007, Singapore, March 20-22, 2007*, pages 204–213, 2007.

[SMCB12] Benedikt Schmidt, Simon Meier, Cas J. F. Cremers, and David A. Basin. Automated analysis of Diffie-Hellman protocols and advanced security properties. In *25th IEEE Computer Security Foundations Symposium, CSF 2012, Cambridge, MA, USA, June 25-27, 2012*, pages 78–94, 2012.

[SSBC09] Patrick Schaller, Benedikt Schmidt, David A. Basin, and Srdjan Capkun. Modeling and verifying physical properties of security protocols for wireless networks. In *Proceedings of the 22nd IEEE Computer Security Foundations Symposium, CSF 2009, Port Jefferson, New York, USA, July 8-10, 2009*, pages 109–123, 2009.

[STBF17] Yamisleydi Salgueiro, Jorge L. Toro, Rafael Bello, and Rafael Falcon. Multi-objective variable mesh optimization. *Annals OR*, 258(2):869–893, 2017.

[SWP09] Amitabh Saxena, Brecht Wyseur, and Bart Preneel. Towards security notions for white-box cryptography. In *Information Security, 12th International Conference, ISC 2009, Pisa, Italy, September 7-9, 2009. Proceedings*, pages 49–58, 2009.

[Tan81] Robert Michael Tanner. A recursive approach to low complexity codes. *IEEE Trans. Information Theory*, 27(5):533–547, 1981.

[TDJM+11] Peter Thueringer, Hans De Jong, Bruce Murray, Heike Neumann, Paul Hubmer, and Susanne Stern. Decoupling of measuring the response time of a transponder and its authentication, March 2011. US Patent App. 12/994,541.

[TMA10] Rolando Trujillo-Rasua, Benjamin Martin, and Gildas Avoine. The Poulidor distance-bounding protocol. In *Proc. 6th International Conference on Radio Frequency Identification: Security and Privacy Issues (RFIDSec'10)*, volume 6370 of *LNCS*, pages 239–257. Springer, 2010.

[TMA14] Rolando Trujillo-Rasua, Benjamin Martin, and Gildas Avoine. Distance bounding facing both mafia and distance frauds. *IEEE Transactions on Wireless Communications*, 13(10):5690–5698, 2014.

[TP07] Yu-Ju Tu and Selwyn Piramuthu. RFID distance bounding protocols. In *Proc. First International EURASIP Workshop on RFID Technology*, pages 67–68, 2007.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91, 1982.

[YKHL11] Dae Hyun Yum, Jin Seok Kim, Sung Je Hong, and Pil Joong Lee. Distance bounding protocol with adjustable false acceptance rate. *Communications Letters, IEEE*, 15(4):434–436, April 2011.

[YPM+18] Anjia Yang, Elena Pagnin, Aikaterini Mitrokotsa, Gerhard P. Hancke, and Duncan S. Wong. Two-hop distance-bounding protocols: Keep your friends close. *IEEE Trans. Mob. Comput.*, 17(7):1723–1736, 2018.

[YY96] Adam L. Young and Moti Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*, pages 89–103, 1996.

[Zém91] Gilles Zémor. Hash functions and graphs with large girths. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, pages 508–511, 1991.