Symbolic analysis of terrorist fraud resistance *

Alexandre Debant[1], Stéphanie Delaune[1], and Cyrille Wiedling[2]

[1] Univ Rennes, CNRS, IRISA, France
{alexandre.debant,stephanie.delaune}@irisa.fr
[2] DGA MI, Bruz, France
cwiedling@gmail.com

Abstract. Distance-bounding protocols aim at preventing several kinds of attacks, amongst which terrorist fraud, where a far away malicious prover colludes with an attacker to authenticate once, without giving him any advantage for future authentication. In this paper, we consider a symbolic setting and propose a formal definition of terrorist fraud, as well as two reduction results. When looking for an attack, we can first restrict ourselves to consider a particular (and quite simple) topology. Moreover, under some mild hypotheses, the far away malicious prover has a best strategy on which we can focus on when looking for an attack. These two reduction results make possible the analysis of terrorist fraud resistance using an existing verification tool. As an application, we analyse several distance-bounding protocols, as well as some contactless payment protocols using the ProVerif tool.

1 Introduction

Contactless devices deployed today in ticketing and building access-control applications are supposed to make our life easier but they also make possible new kinds of attacks, e.g. relay attacks. An attacker can use two transponders (two mobile phones could be sufficient) in order to relay over a large distance the information between e.g. a card and an access card reader. As a result, an unauthorised person will be able to enter a building using an access card located far away and possibly still in the pocket of his holder. With the deployment of contactless systems, ensuring "proximity authentication", through the use of secure protocols, is an important goal.

Relay attacks cannot be prevented by traditional cryptographic protocols. One possible defence is *distance bounding protocols*. The main goal of a distance bounding protocol consists of ensuring that a prover is within a close distance to a verifier by timing the round-trip delay of a cryptographic challenge-response exchange. Therefore the security of these protocols is based on the physical limits of communication: transmission can not go faster than the speed of light. Since they have been introduced by Brands and Chaum in 1993 [8], many protocols

* This work has been partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 714955-POPSTAR).

have been designed and analysed against various threats. In general, distance bounding protocols shall resist to *distance fraud*: a malicious prover should not be able to successfully complete a session with an honest verifier who is far away (even with the help of some honest provers in the neighbourhood - so called *distance hijacking*). They should also resist to *mafia fraud* where typically an attacker abuses a far away prover to pass the protocol. In most cases, this is achieved by relaying messages between the prover and the verifier (the so-called *relay attack*). A more subtle notion is the notion of *terrorist fraud*. Here, a far away malicious prover colludes with an attacker who is close to the verifier to pass the protocol on his behalf. Such a scenario may occur if a legitimate worker want to enable a third party to access his office, located in a restricted area, when he is away. To prevent such behaviours, the protocol is said resistant to a terrorist fraud if this help is actually reusable meaning that the third party can use this extra information to impersonate the prover later on. The rationale is that a malicious prover will not accept to give such an advantage to his accomplice, and thus will not accept to collude with the attacker. This type of attack is very tricky and rather difficult to model and analyse since it requires to consider "terrorist" provers that are not fully dishonest in the sense that they are not willing for instance to reveal their credentials.

Formal symbolic modelling and analysing techniques have proved their usefulness for verifying security protocols, and nowadays several verification tools exist, e.g. ProVerif [5, 6], Tamarin [26]. Since the seminal paper by Dolev-Yao in [16], a lot of progress has been done in the area of formal symbolic verification and it is now a common good practice to formally analyse a protocol using these techniques before their deployment. In this so-called Dolev-Yao model, messages are transmitted without introducing any delay preventing us to use this model to analyse protocols for which transmission delay plays an important role. To overcome this limitation, getting some inspiration from earlier works (e.g. [25, 4, 28]), some recent works have proposed to incorporate new features in existing symbolic models [23, 14, 11, 13], making the analysis of distance bounding protocols possible relying on existing verification tools (e.g. ProVerif, Tamarin).

Our contributions. In this paper, distance bounding protocols are modelled using the calculus we introduced in [14]. This calculus shares some similarities with the applied pi calculus [1, 6], a well-established process algebra for modelling cryptographic protocols. Within this framework, we propose a formal definition of terrorist fraud. We will see that this notion is tricky and complex and require a quantification over all the topologies, but also another one to consider all the possible terrorist provers. Due to this, such a security property can not be analysed using techniques deployed in e.g. [23, 14, 13]. Our main contribution is to provide reduction results to reduce the number of topologies we have to consider during our analysis, and more importantly to reduce the possible behaviours of our terrorist prover. We will see that under some reasonable conditions, we are able to reduce the number of topologies to be considered to one (involving at most 4 participants), and the best strategy for the terrorist prover can also be fixed without missing any attack. Then, an interesting consequence of our

results is that, following the approach used e.g. in [14, 11], it becomes possible to rely on the automatic verification tool ProVerif (originally developed to analyse traditional security protocols) to analyse terrorist fraud in various distance bounding protocols. All the omitted proofs are available in the full version [15].

Related works. Several attempts have been made in the computational model to formalise terrorist fraud, e.g. [17, 2, 18, 30]. Avoine *et al.* [2] introduce a unified framework for clarifying the situation and make possible comparison between protocols. Since then, several formal definitions of terrorist fraud have been proposed [18, 30], as well as protocols supposed to achieve this level of security, e.g. [22, 9, 3]. In contrast, the only definition we are aware of in the symbolic model is the one proposed by Chothia et al in [11]. However, such a definition falls short when modelling behaviours of terrorist provers (see Section 3). Independently of our work, Mauw et al proposed a definition more in line with the one we considered here [24]. However, their work falls short when it comes to the automation of security analysis (see Section 4.3).

2 Model for distance bounding protocols

In this section, we introduce the process calculus we rely on to describe distance bounding protocols [14]. It shares some similarities with the applied pi calculus used e.g. by the ProVerif verification tool [6].

2.1 Messages

As usual in the symbolic setting, we model messages through a term algebra. We consider both equational theories and reduction relations to represent the properties of the cryptographic primitives.

Term algebra. We consider two infinite and disjoint sets of *names*: $\mathcal{N}$ is the set of *basic names*, which are used to represent keys, nonces, whereas $\mathcal{A}$ is the set of *agent names*, *i.e.* names which represent the agents identities. We consider an infinite set Σ_0 of constant symbols that are used to represent values known by the attacker, as well as two infinite and disjoint sets of *variables*, denoted $\mathcal{X}$ and $\mathcal{W}$. Variables in $\mathcal{X}$ refer to unknown parts of messages expected by participants while variables in $\mathcal{W}$ are used to store messages learnt by the attacker.

We assume a signature Σ, *i.e.* a set of function symbols together with their arity. The elements of Σ are split into *constructor* and *destructor* symbols, *i.e.* $\Sigma = \Sigma_c \uplus \Sigma_d$. We denote $\Sigma^+ = \Sigma \cup \Sigma_0$, and $\Sigma_c^+ = \Sigma_c \cup \Sigma_0$. Given a signature $\mathcal{F}$, and a set of atomic data A, we denote by $\mathcal{T}(\mathcal{F}, \mathsf{A})$ the set of *terms* built from atomic data A by applying function symbols in $\mathcal{F}$. A *constructor term* is a term in $\mathcal{T}(\Sigma_c^+, \mathcal{N} \cup \mathcal{A} \cup \mathcal{X})$. We denote $vars(u)$ the set of variables that occur in a term u. A *message* is a constructor term u that is *ground*, *i.e.* such that $vars(u) = \emptyset$. The application of a substitution σ to a term u is written $u\sigma$. We denote $dom(\sigma)$ its *domain*, and $img(\sigma)$ its *image*. The positions of a term are defined as usual.

Example 1. We consider the signature $\Sigma_{\mathsf{ex}} = \{\mathsf{kdf}/3, \mathsf{shk}/2, \mathsf{ok}/0, \mathsf{eq}/2, \mathsf{ans}/3\}$. The symbol kdf models a key derivation function, shk is used to model a key shared between 2 agents. The symbols ok and eq are used to model equality tests, and ans is a function symbol that is used to model the answer provided by the prover. Another signature useful to model the exclusive-or operator is $\Sigma_{\mathsf{xor}} = \{\oplus, 0\}$. Among all the symbols in $\Sigma_{\mathsf{ex}} \cup \Sigma_{\mathsf{xor}}$ only eq is a destructor.

Equational theory. Following the approach developed in [6], constructor terms are subject to an *equational theory* allowing us to model the algebraic properties of the primitives. It consists of a finite set of equations of the form $u = v$ where $u, v \in \mathcal{T}(\Sigma_c, \mathcal{X})$, and induces an equivalence relation $=_{\mathsf{E}}$ over constructor terms.[1]

Example 2. To reflect the algebraic properties of the exclusive-or operator, we may consider the equational theory $\mathsf{E}_{\mathsf{xor}}$ generated by the following equations:

$$(x \oplus y) \oplus z = x \oplus (y \oplus z) \qquad x \oplus y = y \oplus x \qquad x \oplus 0 = x \qquad x \oplus x = 0.$$

Rewriting rules. As in [6], we also give a meaning to destructor symbols. This is done through a set of rewrite rules of the form $\mathsf{g}(t_1, \ldots, t_n) \to t$ where $\mathsf{g} \in \Sigma_d$, and $t, t_1, \ldots, t_n \in \mathcal{T}(\Sigma_c, \mathcal{X})$. A term u can be *rewritten* in v if there is a position p in u, and a rewrite rule $\mathsf{g}(t_1, \ldots, t_n) \to t$ such that $u|_p =_{\mathsf{E}} \mathsf{g}(t_1, \ldots, t_n)\theta$ for some substitution θ, and $v = u[t\theta]_p$ i.e. u in which the term at position p has been replaced by $t\theta$. Moreover, we assume that $t_1\theta, \ldots, t_n\theta$ as well as $t\theta$ are constructor terms. We only consider sets of rewrite rules that yield a *convergent* rewriting system (modulo E), and we denote $u{\downarrow}$ the *normal form* of a term u.

For modelling purposes, we split the signature Σ into two parts, Σ_{pub} and Σ_{priv}, and we denote $\Sigma^+_{\mathsf{pub}} = \Sigma_{\mathsf{pub}} \cup \Sigma_0$. An attacker builds messages by applying public symbols to terms he knows and that are available through variables in $\mathcal{W}$. Formally, a computation done by the attacker is a *recipe*, *i.e.* a term in $\mathcal{T}(\Sigma^+_{\mathsf{pub}}, \mathcal{W})$.

Example 3. Among symbols in $\Sigma_{\mathsf{ex}} \cup \Sigma_{\mathsf{xor}}$, only shk is in Σ_{priv}. The property of the symbol eq is reflected by the rule $\mathsf{eq}(x, x) \to \mathsf{ok}$. Note that $\mathsf{eq}(u, v)$ reduces to a message if, and only if, $u =_{\mathsf{E}} v$. A typical signature used to model security protocols is $\Sigma_{\mathsf{enc}} = \{\mathsf{senc}, \mathsf{sdec}\}$. Depending on whether we want to model a decryption algorithm that may fail or not, we can either consider sdec as a destructor together with the rewrite rule $\mathsf{sdec}(\mathsf{senc}(x, y), y) \to x$, or consider both symbols as constructors, together with equation $\mathsf{sdec}(\mathsf{senc}(x, y), y) = x$. In the latter case, $\mathsf{sdec}(c, k)$ will be considered as a "valid" message.

Given a set $\mathcal{U}$ of equations between terms, σ is a *unifier* for $\mathcal{U}$ if $u_1\sigma{\downarrow} =_{\mathsf{E}} u_2\sigma{\downarrow}$ and both $u_1\sigma{\downarrow}$ and $u_2\sigma{\downarrow}$ are constructor terms for any $u_1 = u_2 \in \mathcal{U}$. We denote by $\mathsf{csu}(\mathcal{U})$ a set of unifiers for $\mathcal{U}$ which is also *complete*, i.e. such that for any σ unifier of $\mathcal{U}$, there exists $\theta \in \mathsf{csu}(\mathcal{U})$ such that $\sigma =_{\mathsf{E}} \tau \circ \theta$ for some τ.

Example 4. Let $\mathcal{U} = \{x_0 = m_0, x_1 = k \oplus x_0; x_{\mathsf{ok}} = \mathsf{eq}(x_{\mathsf{rep}}, \mathsf{ans}(c, x_0, x_1))\}$ with $k = \mathsf{shk}(p_0, v_0)$, and $m_0 = \mathsf{kdf}(k, n_V, x_N)$. We have that $\mathsf{csu}(\mathcal{U}) = \{\theta\}$ where θ is the substitution: $x_0 \mapsto m_0; x_1 \mapsto k \oplus m_0; x_{\mathsf{rep}} \mapsto \mathsf{ans}(c, m_0, k \oplus m_0);\ x_{\mathsf{ok}} \mapsto \mathsf{ok}$.

[1] We only consider non-trivial theories, i.e. there exist u and v such that $u \neq_{\mathsf{E}} v$.

2.2 Protocols

Protocols are modelled through processes that may receive and send messages.

Syntax. We consider the following grammar:

$$P, Q = 0 \mid \mathtt{in}(x).P \mid \mathtt{in}^{<t}(x).P \mid \mathtt{let}\ x = v\ \mathtt{in}\ P \mid \mathtt{new}\ n.P \mid \mathtt{out}(u).P \mid \mathtt{reset}.P$$

where $x \in \mathcal{X}$, $n \in \mathcal{N}$, $u \in \mathcal{T}(\Sigma_c^+, \mathcal{X} \uplus \mathcal{N} \uplus \mathcal{A})$, $v \in \mathcal{T}(\Sigma^+, \mathcal{X} \uplus \mathcal{N} \uplus \mathcal{A})$ and $t \in \mathbb{R}_+$.

We write $fv(P)$ (resp. $fn(P)$) for the set of *free* variables (resp. names) occurring in P, *i.e.* the set of variables (resp. names) that are not in the scope of an input or a let (resp. a new). In this work, we only consider 2-party protocols, and thus we consider *parameterised processes*, denoted $P(z_0, z_1)$, where z_0 and z_1 are variables from a special set $\mathcal{Z}$ (disjoint from $\mathcal{X}$ and $\mathcal{W}$). Intuitively, z_0 and z_1 will be instantiated by agent names: z_0 corresponds to the name of the agent that executes the process, and z_1 will be his interlocutor. A *role* $R = P(z_0, z_1)$ is a parameterised process such that $fn(R) = \emptyset$ and $fv(R) \subseteq \{z_0, z_1\}$. A *protocol* is given by two roles, denoted $\mathtt{V}(z_0, z_1)$ and $\mathtt{P}(z_0, z_1)$, and named respectively the *verifier role* and the *prover role*. Moreover, we will assume that the verifier role ends with a special construct $\mathsf{end}(z_0, z_1)$ allowing us to see when he has completed his role and with whom. Formally, it simply means that, in the verifier role $\mathtt{V}(z_0, z_1)$, the process 0 has been replaced by $\mathsf{end}(z_0, z_1)$.

Example 5. As a running example and for illustrative purposes, we consider a strengthened version of the Hancke and Kuhn distance bounding protocol [20] (as briefly described in [29]). It relies on the use of a keyed public pseudo-random function (modelled as a free function symbol here) and the exclusive-or operator.

1. $V \to P$: N_V
2. $P \to V$: N_P
3. $V \to P$: c_i
4. $P \to V$: $\begin{cases} i^{\text{th}} \text{ bit of } \mathsf{kdf}(k, N_V, N_P) \text{ if } c_i = 0 \\ i^{\text{th}} \text{ bit of } k \oplus \mathsf{kdf}(k, N_V, N_P) \text{ if } c_i = 1 \end{cases}$

The protocol starts with both parties transmitting to each other their own nonce. Then, the verifier initiates the rapid phase during which the time measurement is performed. The verifier generates and sends a random bit c_i, and the prover has to reply immediately with the i^{th} bit of $\mathsf{kdf}(k, N_V, N_P)$ if $c_i = 0$ and the i^{th} bit of $k \oplus \mathsf{kdf}(k, N_V, N_P)$ otherwise. This rapid exchange is repeated a fixed number of times, and if enough correct answers are received within a sufficiently short time after the corresponding challenge c_i has been sent out, then the verifier is convinced that the prover is located in its vicinity. In our setting, this gives us:

$\mathtt{V}(z_0, z_1) :=$
$\quad \mathtt{new}\ n_V.\mathtt{out}(n_V).\mathtt{in}(x_N).$
$\quad \mathtt{reset}.\mathtt{new}\ c.\mathtt{out}(c).\mathtt{in}^{<2\times t_0}(x_{\mathsf{rep}}).$
$\quad \mathtt{let}\ x_0 = \mathsf{kdf}(\mathsf{shk}(z_1, z_0), n_V, x_N)\ \mathtt{in}$
$\quad \mathtt{let}\ x_1 = \mathsf{shk}(z_1, z_0) \oplus x_0\ \mathtt{in}$
$\quad \mathtt{let}\ x_{ok} = \mathsf{eq}(x_{\mathsf{rep}}, \mathsf{ans}(c, x_0, x_1))\ \mathtt{in}$
$\quad \mathsf{end}(z_0, z_1)$

$\mathtt{P}(z_0, z_1) :=$
$\quad \mathtt{new}\ n_P.\mathtt{in}(y_N).\mathtt{out}(n_P).$
$\quad \mathtt{let}\ y_0 = \mathsf{kdf}(\mathsf{shk}(z_0, z_1), y_N, n_P)\ \mathtt{in}$
$\quad \mathtt{let}\ y_1 = \mathsf{shk}(z_0, z_1) \oplus y_0\ \mathtt{in}$
$\quad \mathtt{in}(y_c).$
$\quad \mathtt{out}(\mathsf{ans}(y_c, y_0, y_1)).0$

Symbolic analysis does not allow one to reason at the bit level, and thus, as done in e.g. [23, 14, 11, 13], all the challenge bits c_i are collapsed into a single challenge/response exchange using a nonce. Furthermore, operations performed at the bit level are abstracted too. The response is therefore abstracted by an uninterpreted symbol of a function ans depending on both the challenge c and the two precomputed values y_0 and y_1.

Topology. The semantics of our processes depends on their location. This is formally defined through the notion of topology.

Definition 1. *A* topology *is a tuple* $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ *where:*

- $\mathcal{A}_0 \subseteq \mathcal{A}$ *is the finite set of agents composing the system;*
- $\mathcal{M}_0 \subseteq \mathcal{A}_0$ *is the subset of agents that are malicious;*
- $\mathsf{Loc}_0 : \mathcal{A}_0 \to \mathbb{R}^3$ *is a mapping defining the position of each agent in space.*
- p_0 *and* v_0 *are two agents in* $\mathcal{A}_0$ *that represent respectively the prover and the verifier w.r.t. which the analyse is performed.*

In our model, the distance between two agents is expressed by the time it takes for a message to travel from one to another. Therefore, we consider $\mathsf{Dist}_{\mathcal{T}_0} : \mathcal{A}_0 \times \mathcal{A}_0 \to \mathbb{R}$, based on Loc_0 that will provide the time a message takes to travel between two agents. It is defined as follows:

$$\mathsf{Dist}_{\mathcal{T}_0}(a, b) = \frac{\|\mathsf{Loc}_0(a) - \mathsf{Loc}_0(b)\|}{c_0} \text{ for any } a, b \in \mathcal{A}_0$$

with $\|\cdot\| : \mathbb{R}^3 \to \mathbb{R}$ the Euclidean norm and c_0 the transmission speed. We suppose, from now on, that c_0 is a constant for all agents, and thus an agent a can recover, at time t, any message emitted by any other agent b before $t - \mathsf{Dist}_{\mathcal{T}_0}(a, b)$.

Example 6. When analysing a distance bounding protocol, we have to consider a class of topologies. Typically, a mafia fraud is an attack in which at least three agents are involved: an honest verifier, an honest prover, and an attacker. Of course, in general more agents may be involved, and the set $\mathcal{C}_{\mathsf{MF}}$ of all the mafia fraud topologies is simply defined as follows: any topology $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ such that $v_0, p_0 \in \mathcal{A}_0 \setminus \mathcal{M}_0$.

Configuration. The semantics of our processes is given through a transition system defined over configurations. Given a topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$, a *configuration* K over $\mathcal{T}_0$ is a tuple $(\mathcal{P}; \Phi; t)$, where:

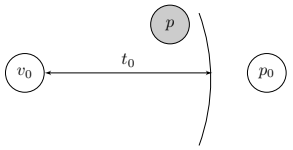
- $\mathcal{P}$ is a multiset of *extended process* $\lfloor P \rfloor_a^{t_a}$ with $a \in \mathcal{A}_0$ and $t_a \in \mathbb{R}_+$;
- $\Phi = \{\mathsf{w}_1 \xrightarrow{a_1, t_1} u_1, \ldots, \mathsf{w}_n \xrightarrow{a_n, t_n} u_n\}$ is an extended frame, i.e. a substitution such that $\mathsf{w}_i \in \mathcal{W}$, $u_i \in \mathcal{T}(\Sigma_c^+, \mathcal{N} \uplus \mathcal{A})$, $a_i \in \mathcal{A}_0$ and $t_i \in \mathbb{R}_+$ for $1 \leq i \leq n$;
- $t \in \mathbb{R}_+$ is the global time of the system.

A *initial frame* is a frame such that $t_i = 0$ $(1 \leq i \leq n)$, and an *initial configuration* is a configuration such that $t = 0$. We write $\lfloor \Phi \rfloor_a^t$ for the restriction of Φ to the agent a at time t, *i.e.* :

$$\lfloor \Phi \rfloor_a^t = \left\{ \mathsf{w}_i \xrightarrow{a_i, t_i} u_i \mid (\mathsf{w}_i \xrightarrow{a_i, t_i} u_i) \in \Phi \text{ and } a_i = a \text{ and } t_i \leq t \right\}.$$

Example 7. Continuing Example 5, we consider $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ depicted below where $\mathcal{A}_0 = \{p_0, v_0, p\}$, and $\mathcal{M}_0 = \{p\}$.

The precise location of each agent is not relevant, only the distance between them matters. Here $\mathsf{Dist}_{\mathcal{T}_0}(p, v_0) < t_0$ whereas $\mathsf{Dist}_{\mathcal{T}_0}(p_0, v_0) \geq t_0$.



A possible initial configuration K_0 is given below:

$$(\lfloor \mathtt{P}(p_0, v_0) \rfloor_{p_0}^{0} \uplus \lfloor \mathtt{V}(v_0, p_0) \rfloor_{v_0}^{0} ; \{\mathsf{w}_1 \xrightarrow{p,0} \mathsf{shk}(p, v_0), \mathsf{w}_2 \xrightarrow{p,0} m_0, \mathsf{w}_3 \xrightarrow{p,0} m_1\}; 0)$$

Here, p_0 and v_0 are honest agents playing respectively the prover's role and the verifier's role. The agent p is a malicious prover whose shared key with v_0 is given to the attacker through w_1. Here, we also assume that the attacker p also knows $m_0 = \mathsf{kdf}(\mathsf{shk}(p_0, v_0), n_V^0, n_P^0)$ and $m_1 = \mathsf{shk}(p_0, v_0) \oplus \mathsf{kdf}(\mathsf{shk}(p_0, v_0), n_V^0, n_P^0)$. These messages coming from an older session may have been given to him by p_0 to let the attacker exceptionally authenticate on his behalf. A more realistic configuration will include other instances of these two roles and will probably give more knowledge to the attacker.

Semantics. We now recall the semantics of our calculus as defined in [14].

TIM $\quad (\mathcal{P}; \Phi; t) \longrightarrow_{\mathcal{T}_0} (\mathsf{Shift}(\mathcal{P}, \delta); \Phi; t + \delta)$ with $\delta \geq 0$

RST $\quad (\lfloor \mathtt{reset}.P \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P \rfloor_a^{0} \uplus \mathcal{P}; \Phi; t)$

OUT $\quad (\lfloor \mathtt{out}(u).P \rfloor_a^{t_a}) \uplus \mathcal{P}; \Phi; t) \xrightarrow{a,\mathtt{out}(u)}_{\mathcal{T}_0} (\lfloor P \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi \uplus \{\mathsf{w} \xrightarrow{a,t} u\}; t)$ with $\mathsf{w} \in \mathcal{W}$ fresh

LET $\quad (\lfloor \mathtt{let}\ x = u\ \mathtt{in}\ P \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P\{x \mapsto u{\downarrow}\} \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t)$ when $u{\downarrow} \in \mathcal{T}(\Sigma_c^+, \mathcal{N} \uplus \mathcal{A})$

NEW $\quad (\lfloor \mathtt{new}\ n.P \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P\{n \mapsto n'\} \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t)$ with $n' \in \mathcal{N}$ fresh

IN $\quad (\lfloor \mathtt{in}^{\star}(x).P \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t) \xrightarrow{a,\mathtt{in}^{\star}(u)}_{\mathcal{T}_0} (\lfloor P\{x \mapsto u\} \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t)$

when there exist $b \in \mathcal{A}_0$ and $t_b \in \mathbb{R}_+$ such that $t_b \leq t - \mathsf{Dist}_{\mathcal{T}_0}(b, a)$ and:

- if $b \in \mathcal{A}_0 \setminus \mathcal{M}_0$ then $u \in img(\lfloor \Phi \rfloor_b^{t_b})$;
- if $b \in \mathcal{M}_0$ then $u = R\Phi{\downarrow}$ for some recipe R such that for all $\mathsf{w} \in vars(R)$ there exists $c \in \mathcal{A}_0$ such that $\mathsf{w} \in dom(\lfloor \Phi \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}_0}(c,b)})$.

Moreover, in case $\star$ is $< t_g$ for some t_g, we assume in addition that $t_a < t_g$.

The two first rules are specific to our timed model. The RST rule allows a process to reset its local clock, whereas the TIM rule allows time to elapse, meaning that all the clocks will be shifted by δ:

$$\mathsf{Shift}(\mathcal{P}, \delta) = \biguplus_{\lfloor P \rfloor_a^{t_a} \in \mathcal{P}} \mathsf{Shift}(\lfloor P \rfloor_a^{t_a}, \delta) \text{ and } \mathsf{Shift}(\lfloor P \rfloor_a^{t_a}, \delta) = \lfloor P \rfloor_a^{t_a + \delta}.$$

The remaining rules are quite standard. The Out rule is used to output a message which is immediately added into the frame. The rule Let can be used to apply function symbols, e.g. $\texttt{let}\ x = \mathsf{dec}(y,k)\ \texttt{in}\ P$ applies decryption on top of y with the key k and store the resulting result in x (if this operation succeeds). Otherwise, the process is blocked. This construction is also useful to perform equality tests through the symbol eq as defined in Example 1 and used e.g. in Example 5. The New rule allows one to pick a fresh (i.e. previously unused) name. Finally, the In rule is used to receive a message. One can note the additional side conditions which allows one to model timing constraints: all the messages needed to construct u have to be available to b (who sends u) at time $t_b \leq t - \mathsf{Dist}_{\mathcal{T}_0}(b,a)$ to ensure that the message forged and sent by b will have enough time to travel and reach a.

Example 8. To illustrate our semantics, we give below a possible execution trace starting from the configuration K_0 given in Example 7. We have that:

$$K_0 \xrightarrow{(v_0,\tau).(v_0,\texttt{out}(n_V))} \mathcal{T}_0 \rightarrow_{\mathcal{T}_0} \xrightarrow{(v_0,\texttt{in}(n_I)).(v_0,\texttt{reset}).(v_0,\tau).(v_0,\texttt{out}(c))} \mathcal{T}_0 \ \ (\mathcal{P}';\Phi';t')$$
$$\rightarrow_{\mathcal{T}_0} \xrightarrow{(v_0,\texttt{in}(m_{\mathsf{rep}})).(v_0,\tau).(v_0,\tau).(v_0,\tau)} \mathcal{T}_0 \ \ (\ \lfloor \mathsf{P}(p_0,v_0)\rfloor_{p_0}^{t''} \uplus \lfloor \mathsf{end}(v_0,p_0)\rfloor_{v_0}^{t''-t'};\Phi'';t'')$$

with $m_{\mathsf{rep}} = \mathsf{ans}(c, \mathsf{kdf}(\mathsf{shk}(p_0,v_0), n_V, n_I), \mathsf{shk}(p_0,v_0) \oplus \mathsf{kdf}(\mathsf{shk}(p_0,v_0), n_V, n_I))$, $t' \geq \mathsf{Dist}_{\mathcal{T}_0}(v_0,p)$, $t'' \geq 3\mathsf{Dist}_{\mathcal{T}_0}(v_0,p)$, $\Phi'' = \Phi' = \Phi_0 \uplus \{\mathsf{w}_4 \xrightarrow{v_0,0} n_V,\ \mathsf{w}_5 \xrightarrow{v_0,t'} c\}$.

Here, n_I is a name known to the attacker. Formally, we have that $n_I \in \Sigma_0$. During the first part of the execution (1^{st} line), one instance of the Tim rule has been used. It is necessary to let the verifier receive n_I. Therefore, we have that $t' \geq \mathsf{Dist}_{\mathcal{T}_0}(v_0,p)$. The attacker has learnt two messages that have been added into his initial frame Φ_0. Then, letting some time to elapse, the process located in v_0 is able to perform his input action. Indeed, the term m_{rep} can be forged by p using recipe $\mathsf{ans}(\mathsf{w}_5, R_0, R_1)$ where $R_0 = \mathsf{kdf}(\mathsf{w}_2 \oplus \mathsf{w}_3, \mathsf{w}_4, n_I)$ and $R_1 = \mathsf{w}_2 \oplus \mathsf{w}_3 \oplus R_0$. We may note that m_{rep} passes successfully all the tests, and v_0 ends his session thinking he is talking to p_0 (who is actually far away).

3 Modelling Mafia and Terrorist Frauds

Here, we aim at proposing a general definition of terrorist fraud in the symbolic setting. Due to its close relationship with the notion of mafia fraud, we first recall how mafia fraud is modelled following the definitions given in [14] before defining the more subtle notion of terrorist fraud.

We start by defining the notion of valid initial configurations which corresponds to the configurations that need to be studied when analysing a given protocol $\mathcal{P}$. Typically, such a configuration will contain instances of the roles of the protocol $\mathcal{P}$ under study.

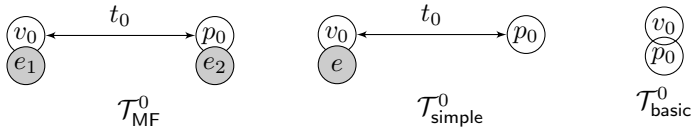
Definition 2. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology, and Φ_0 be an initial frame. $K = (\mathcal{P};\Phi;t)$ is a* valid initial configuration *for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}_0$ and Φ_0 if $t = 0$, $\Phi = \Phi_0$, and for each $\lfloor P'\rfloor_{a'}^{t'} \in \mathcal{P}$, we have that $t' = 0$, $a' \in \mathcal{A}_0$, and either $P' = \mathtt{V}(a',b')$ or $P' = \mathtt{P}(a',b')$ for some $b' \in \mathcal{A}_0$.*

Now, depending on the type of frauds we consider, the set of topologies under study and the initial knowledge given to the attacker may vary.

3.1 Mafia fraud

A *mafia fraud* is an attack in which generally three agents are involved: a verifier, an honest prover located outside the neighbourhood of the verifier, and an attacker. We consider here its general version which may involve an arbitrary number of participants and we reuse the definition given in [14]. The aim of the attacker is to convince the verifier that the honest prover is actually close to it. The set $\mathcal{C}_{\mathsf{MF}}$ representing all the mafia fraud topologies is given in Example 6.

Example 9. The topology depicted in Example 7 is a mafia fraud topology. Some other mafia fraud topologies that will be considered later on are depicted below:



The initial knowledge Φ_0 we use for defining our initial configuration depends on the topology but it is reasonable to assume that this knowledge is uniform. Therefore, we assume that the initial knowledge of all the participants is given through a template $\mathcal{I}_0$, i.e. a set of terms in $\mathcal{T}(\Sigma_c^+, \{z_0, z_1\})$. Relying on $\mathcal{I}_0$, and considering a set $\mathcal{A}_0$ of agents, the initial knowledge of agent $a \in \mathcal{A}_0$ is given by:

$$\mathsf{Knows}_{\mathcal{I}_0}(a, \mathcal{A}_0) = \{u_0\{z_0 \mapsto a, z_1 \mapsto b\} \mid u_0 \in \mathcal{I}_0 \text{ and } b \in \mathcal{A}_0\}$$

Given $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$, we denote $\Phi^{\mathcal{T}}_{\mathcal{I}_0}$ the initial frame such that $\lfloor img(\Phi^{\mathcal{T}}_{\mathcal{I}_0})\rfloor^{\,0}_{a} = \mathsf{Knows}_{\mathcal{I}_0}(a, \mathcal{A}_0)$ when $a \in \mathcal{M}_0$, and $\lfloor img(\Phi^{\mathcal{T}}_{\mathcal{I}_0})\rfloor^{\,0}_{a} = \emptyset$ otherwise. Up to a renaming of the handles and some duplicates, $\Phi^{\mathcal{T}}_{\mathcal{I}_0}$ is uniquely defined.

Example 10. Continuing our running example, and considering the topology $\mathcal{T}^0_{\mathsf{MF}} = (\mathcal{A}^0_{\mathsf{MF}}, \mathcal{M}^0_{\mathsf{MF}}, \mathsf{Loc}^0_{\mathsf{MF}}, v_0, p_0)$ (see Example 9), a typical template to derive the initial knowledge of the malicious agents is $\mathcal{I}_0 = \{\mathsf{shk}(z_0, z_1), \mathsf{shk}(z_1, z_0)\}$. Thus, considering the malicious agent e_i, the set $\mathsf{Knows}_{\mathcal{I}_0}(e_i, \mathcal{A}^0_{\mathsf{MF}})$ will contain all the symmetric keys this malicious agent shares with other agents.

When analysing the protocol considering $\mathcal{T}^0_{\mathsf{MF}}$, we will consider an initial frame Φ_0 containing $\mathsf{shk}(a, b)$ for $(a, b) \in (\mathcal{A}^0_{\mathsf{MF}} \times \mathcal{A}^0_{\mathsf{MF}}) \setminus \{(v_0, p_0), (p_0, v_0)\}$.

Definition 3. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol and $\mathcal{I}_0$ be a template. We say that $\mathcal{P}_{\mathsf{prox}}$* admits a mafia fraud *w.r.t. t_0-proximity if there exist $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$, and a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^{\mathcal{T}}_{\mathcal{I}_0}$ such that:*

$$K \to^*_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0)\rfloor^{\,t_a}_{a} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathit{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0$$

where $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$.

3.2 Terrorist fraud

Modelling terrorist fraud is tricky. We have to look for a semi-dishonest prover who colludes with the attacker with the aim of letting him authenticate exactly once. We will model this in two steps. First, we will consider all the possible behaviours for this semi-dishonest prover that will allow the attacker to authenticate at least once. Then, to be terrorist fraud resistant, we have to check that any of these behaviours will allow the attacker to re-authenticate later on.

In order to collude with the attacker, one possibility for the prover is to leak his credentials but it is in general not the only option. To define this notion, we consider a simple scenario where p_0 wants to authenticate to the far away verifier v_0 through the help of the attacker a located in the neighbourhood of v_0. This corresponds to the topology $\mathcal{T}^0_{\mathsf{simple}}$ given in Example 9.

Definition 4. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol and $t_0 \in \mathbb{R}_+$. A* semi-dishonest prover *for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. t_0 is a process P_{sd} together with an initial frame Φ_{sd} such that:*

$$(\{\lfloor \mathtt{V}(v_0, p_0)\rfloor^0_{v_0}\,;\, \lfloor P_{\mathsf{sd}}\rfloor^0_{p_0}\}; \emptyset; 0) \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{simple}}} (\{\lfloor \mathsf{end}(v_0, p_0)\rfloor^{t_v}_{v_0}\,;\, \lfloor 0\rfloor^{t_p}_{p_0}\}; \Phi; t)$$

for some t, t_v, t_p, and Φ such that Φ and Φ_{sd} coincide up to their timestamps.

Note that a semi-dishonest prover can be completely dishonest in the sense that he may leak all his credentials. However, such a semi-dishonest prover can not be honest, i.e. equal to the role of the prover as indicated by the protocol. Indeed, p_0 is located far away and has to authenticate. Thus, unless the protocol is very bad, the help of the attacker who is close to the verifier will be essential.

Example 11. Going back to our running example, some semi-dishonest provers with their frame are ($k = \mathsf{shk}(v_0, p_0)$, $m_0 = \mathsf{kdf}(k, n^0_V, n^0_P)$, and $m_1 = k \oplus m_0$):

1. $P^1_{\mathsf{sd}} := \mathtt{out}(k)$ with $\Phi^1_{\mathsf{sd}} = \{\mathsf{w}_1 \xrightarrow{v_0,0} n_V, \mathsf{w}_2 \xrightarrow{p_0,0} k, \mathsf{w}_3 \xrightarrow{v_0,0} c\}$;
2. $P^2_{\mathsf{sd}} := \mathtt{new}\ n_P.\mathtt{in}(y_N).\mathtt{out}(n_P).\mathtt{let}\ y_0 = \mathsf{kdf}(k, y_N, n_P)\ \mathtt{in}$
$\mathtt{let}\ y_1 = k \oplus y_0\ \mathtt{in}\ \mathtt{out}(y_0).\mathtt{out}(y_1).$
$\mathtt{in}(y_c).\mathtt{out}(\mathsf{ans}(y_c, y_0, y_1))$
with $\Phi^2_{\mathsf{sd}} = \{\mathsf{w}_1 \xrightarrow{v_0,0} n_V, \mathsf{w}_2 \xrightarrow{p_0,0} n_P, \mathsf{w}_3 \xrightarrow{p_0,0} m_0, \mathsf{w}_4 \xrightarrow{p_0,0} m_1, \mathsf{w}_5 \xrightarrow{v_0,0} c,$
$\mathsf{w}_6 \xrightarrow{p_0,0} \mathsf{ans}(c, m_0, m_1)\}$.

The first one actually reveals all his credential to the attacker, and thus the attacker will be able to authenticate later on. The second one reveals less information to the attacker. This is still enough to authenticate once and we will see that this is actually also enough to authenticate later on (see Example 12).

We are now able to define our notion of terrorist fraud resistance. Intuitively, if the dishonest prover gives to his accomplice enough information to pass authentication once, then he will be able to authenticate again without his help.

Definition 5. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol and $\mathcal{I}_0$ be a template. We say that $\mathcal{P}_{\mathsf{prox}}$ is* terrorist fraud resistant *w.r.t. t_0-proximity if for all semi-dishonest prover P_{sd}*

with frame Φ_{sd}*, there exist* $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$*, a valid initial configuration* K *for* $\mathcal{P}_{\mathsf{prox}}$ *w.r.t.* $\mathcal{T}$ *and* $\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}$ *such that:*

$$K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathit{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0$$

where $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathit{Loc}_0, v_0, p_0)$.

Example 12. Going back to our running example, we have seen (see Example 8):

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi; t).$$

This execution witnesses the fact that the dishonest prover P^2_{sd} together with frame Φ^2_{sd} gives enough information to the attacker to allow him to authenticate later on. This does not mean that the protocol is terrorist fraud resistant since we only consider one particular semi-dishonest prover. To be terrorist fraud resistant, the property has to hold for *any* semi-dishonest prover.

In our setting, we have the following relationship between mafia fraud and terrorist fraud resistance

Proposition 1. *Let* $\mathcal{P}_{\mathsf{prox}}$ *be a protocol and* $\mathcal{I}_0$ *be a template. If* $\mathcal{P}_{\mathsf{prox}}$ *admits a mafia fraud then* $\mathcal{P}_{\mathsf{prox}}$ *is terrorist fraud resistant (w.r.t.* t_0*-proximity).*

Indeed, whatever the distant semi-dishonest prover discloses (even no information at all), an attacker can still carry out the existing mafia fraud and reauthenticate, therefore impersonating the semi-dishonest prover again. In computational definitions, probability plays a role. In such a setting, a terrorist fraud exists when the semi-dishonest prover can help the attacker to maximise his attack success probability without giving him any advantage for future attacks. The fact that a mafia fraud already exists (with probability 1 in our setting) means that no help can improve the success probability for future attacks, and thus the protocol is terrorist fraud resistant. We may note that distance-bounding protocols designed to achieve terrorist fraud resistance aim also to resist against mafia fraud, thus in general achieving terrorist fraud resistance making the protocol vulnerable to a mafia fraud is not an interesting option.

3.3 Related works

Up to our knowledge, the only existing definitions of terrorist fraud resistance in the symbolic setting are the one proposed by Chothia *et al* in [11] and the recent one proposed by Mauw *et al* in [24].

Chothia et al. Their notion of terrorist fraud is not modelled in two steps as we proposed. Instead, they consider a notion of terrorist prover. Such a process will perform operations on behalf of the attacker, e.g. encrypting and decrypting any values the attacker wishes, but it will never (at least directly) reveal his secrets. Their notion of terrorist prover is appealing but they do not explain how to write such a process. We think that writing such a process is not that easy.

Example 13. Consider a protocol relying on a hash function $\mathtt{h}$ and we assume that the terrorist prover holds a secret key k. A legitimate help that the terrorist prover may give to the attacker without leaking his secret key would consist in computing the hash value of a public data together with his secret key k. Therefore, the terrorist prover should contain the oracle: $\mathtt{in}(x).\mathtt{out}(\mathtt{h}(\langle k, x\rangle))$. In the same spirit, we could argue that $\mathtt{in}(x).\mathtt{out}(\mathtt{h}(\langle x, k\rangle))$ is also useful, and perhaps also $\mathtt{in}(x_1).\mathtt{in}(x_2).\mathtt{out}(\mathtt{h}(\langle x_1, \langle k, x_2\rangle\rangle))$, etc. Iterating such a reasoning, we do not see how to write a finite terrorist prover that will provide all the valuable help his accomplice may need.

Moreover, when considering a protocol involving an operator with some algebraic properties, e.g. exclusive-or, it seems difficult (perhaps even impossible) to ensure that the terrorist prover will not reveal secrets indirectly).

Example 14. To illustrate this issue, we consider a specific primitive modelled using the equation $\mathtt{g}(\mathtt{f}_1(x, y), \mathtt{f}_2(x, y)) = y$. The functions $\mathtt{f}_1$ and $\mathtt{f}_2$ are two constructor symbols whereas $\mathtt{g}$ is a destructor symbol. Following the idea developed in [11], the terrorist prover should contain $\mathtt{in}(x).\mathtt{out}(\mathtt{f}_1(x, k))$ as well as $\mathtt{in}(x).\mathtt{out}(\mathtt{f}_2(\langle x, k\rangle))$. However, whereas it is legitimate to provide such an help to the attacker, it seems too strong to give him access to these oracles as soon as he will get $\mathtt{f}_1(m, k)$ and $\mathtt{f}_2(m_2, k)$ for some message m. This example clearly shows that, combining two legitimate helps, the attacker may retrieve some secrets. It is therefore not obvious to describe in a syntactic way the help the terrorist prover is willing to provide.

The main advantage of the definition of terrorist fraud proposed by [11] is probably the fact that it is more amenable to automation using existing verification tools. Indeed, even if the choice of terrorist prover mentioned in [11] is quite debatable, it is fixed, and can therefore be given in input to the verification tool.

Mauw et al. They consider a model based on multiset rewriting rules, and their definition of terrorist fraud is more in line with the one we proposed. In particular, their notion of "valid extensions" of a protocol seems to correspond to our notion of semi-dishonest prover. Then, their definition of terrorist fraud quantifies over all the possible "valid extensions" and this renders the automation of the security analysis difficult. Indeed, no existing verification tool is able to handle this quantification. In [24], they simply illustrate their technique on a toy distance bounding protocol. They provide a manual proof explaining how to get rid of this quantification for this particular toy protocol. Our work explains in a more systematic way how to get rid of this quantification, as well as the one regarding the topology. This is explained in Section 4.

4 Reduction results

We first establish a result allowing us to focus on a particular topology. Then, we explain how we get rid of the quantification over semi-dishonest provers.

4.1 One topology is enough

This reduction result regarding the topology is a direct consequence of the proof of the reduction result stated in [14] regarding mafia and distance hijacking frauds. The only new issue here is to take care of the initial frame which contains information from the semi-dishonest prover. This reduction results holds in a rather general setting. We simply assume that the protocol under study is executable.

Definition 6. *Given a template $\mathcal{I}_0 = \{u_1, \ldots, u_k\}$, a protocol $\mathcal{P}$ is $\mathcal{I}_0$-executable if for any term u (resp. v) occurring in an* out *or a* let *construction in $\mathcal{P}$, there exists a recipe $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, \{\mathsf{w}_1, \ldots, \mathsf{w}_k\} \uplus \mathcal{N} \uplus \mathcal{X})$ such that $u = R\sigma\downarrow$ (resp. $v\downarrow = R\sigma\downarrow$) where $\sigma = \{\mathsf{w}_1 \mapsto u_1, \ldots, \mathsf{w}_k \mapsto u_k\}$.*

Example 15. Going back to our running example described in Example 5, we have that both roles are $\mathcal{I}_0$-executable considering $\mathcal{I}_0 = \{\mathsf{shk}(z_0, z_1), \mathsf{shk}(z_1, z_0)\}$.

We are now able to state our reduction result regarding terrorist fraud.

Theorem 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be an $\mathcal{I}_0$-executable protocol w.r.t. some template $\mathcal{I}_0$. We have that $\mathcal{P}_{\mathsf{prox}}$ is terrorist fraud resistant w.r.t. t_0-proximity, if and only if, for all semi-dishonest prover P_{sd} with frame Φ_{sd}, there exists a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}^0_{\mathsf{MF}}$ and $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}$ such that:*

$$K \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi; t).$$

In other words, when analysing terrorist fraud, it is sufficient to consider one particular topology, namely $\mathcal{T}^0_{\mathsf{MF}}$ (see Example 9). The key idea to establish the direct part of the theorem consists in showing that behaviours of agents other than p_0 and v_0 can be performed by processes executed by malicious agents, and can even be discarded relying on the fact that $\mathcal{P}_{\mathsf{prox}}$ is $\mathcal{I}_0$-executable. Then, it remains to map any agent names different from p_0 and v_0 to e_1, and to show that the resulting trace remains executable.

4.2 One semi-dishonest prover behaviour is enough

Our second reduction result allows us to focus on a particular semi-dishonest prover when performing our analysis. This results only holds under some hypotheses that are gathered below. We have to rely on the notion of being *quasi-free* for a symbol: $\mathsf{f} \in \Sigma_c$ is *quasi-free* if it occurs neither in the equations used to generate the relation $=_{\mathsf{E}}$ nor in the right-hand side of a rewriting rule.

Definition 7. *A distance bounding (DB) protocol is a protocol such that:*

(i) *We have that $\mathtt{V}(z_0, z_1) = \mathtt{block}_V.\mathtt{reset}.\mathtt{new}\ c.\mathtt{out}(c).\mathtt{in}^{<2\times t_0}(x).\mathtt{block}'_V$, and $\mathtt{P}(z_0, z_1) = \mathtt{block}_P.\mathtt{in}(y_c).\mathtt{out}(u).\mathtt{block}'_P$ where $\mathtt{block}_X$ and $\mathtt{block}'_X$ with $X \in \{V, P\}$ is a sequence of actions without reset and guarded input instructions. Moreover, we assume that $\mathtt{out}(c)$ (resp. $\mathtt{in}(y_c)$) corresponds to the $i_0{}^{th}$ communication action of $\mathtt{P}(z_0, z_1)$ (resp. $\mathtt{V}(z_0, z_1)$) for some i_0.*

(ii) $(\lfloor \mathtt{V}(v_0,p_0)\rfloor^0_{v_0} \uplus \lfloor \mathtt{P}(p_0,v_0)\rfloor^0_{p_0}; \emptyset; 0) \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{basic}}} (\lfloor 0\rfloor^0_{v_0} \uplus \lfloor 0\rfloor^0_{p_0}; \Phi; 0)$ *with*

$$\mathsf{tr} = \begin{cases} (a_1, \mathtt{out}(m_1)).(b_1, \mathtt{in}(m_1)) \dots (a_{i_0-1}, \mathtt{out}(m_{i_0-1})).(b_{i_0-1}, \mathtt{in}(m_{i_0-1})) \\ (v_0, \mathtt{out}(m_{i_0})).(p_0, \mathtt{in}(m_{i_0})).(p_0, \mathtt{out}(m_{i_0+1})).(v_0, \mathtt{in}^{<t}(m_{i_0+1})) \\ (a_{n-1}, \mathtt{out}(m_{n-1}).(b_{n-1}, \mathtt{in}(m_{n-1})) \dots (a_n, \mathtt{out}(m_n).(b_n, \mathtt{in}(m_n)) \end{cases}$$

up to τ actions, and $\{a_i, b_i\} = \{v_0, p_0\}$ for any $i \in \{1, \dots, n\} \setminus \{i_0, i_0+1\}$.

(iii) *Let $\mathcal{U} = \{x = u \mid$ "`let` $x = u$ `in` " *occurs in* $\mathtt{V}(v_0,p_0)\}$. We assume that $\mathsf{csu}(\mathcal{U})$ exists and is reduced to a singleton $\{\theta_{\mathcal{P}}\}$. Moreover, we assume that $(x_1, \dots, x_k)\theta_{\mathcal{P}}{\downarrow}\sigma = m_{i_1}, \dots, m_{i_k}$ where $x_1, \dots, x_k$ are the variables occurring in input in the role $\mathtt{V}(v_0,p_0)$, $i_1, \dots, i_k$ are the indices among $1, \dots, n$ corresponding to input performed by v_0, and σ is a bijective renaming from variables to names freshly generated by $\mathtt{P}(p_0,v_0)$ when executing* tr.

(iv) *We assume the existence of a context C made of quasi-free public function symbols such that $u = C[y_c, u_1, \dots, u_p]$, and y_c does not occur in $u_1, \dots, u_p$.*

The two first conditions put some restrictions on the shape of the roles. In particular, we assume that if no attacker interferes, these two roles together will execute until the end. The third condition gives us the existence of a unique most general unifier (modulo E) and is actually satisfied by many term algebra of interest for protocol verification, e.g. the one described in Section 2.1. Actually, any rewriting system with only one rule per destructor will satisfy such an hypothesis. It may seem restrictive that in a normal execution messages that are exchanged have the shape indicated by $\theta_{\mathcal{P}}$ but this requirement is in general always satisfied. Note that otherwise, it would mean that some terms sent by the prover are never entirely checked during the protocol execution, and thus are useless. Condition *(iv)* allows us to ensure that there exists a strategy for the semi-dishonest prover. This strategy will consist of sending the terms $u_1, \dots, u_n$ in advance to his accomplice, and let him to compute (as indicated by C) the answer to the challenge from $u_1, \dots, u_n$ and the challenge c' he will receive from the verifier. Actually, the best strategy will consist in considering $C_{\mathcal{P}}$ the smallest context (in terms of number of symbols) satisfying the requirements.

Example 16. Going back to our running example, all the conditions stated in Definition 7, are indeed satisfied. Assuming that names are not renamed when executing New, we obtain the following trace:

$$\mathsf{tr} = \begin{cases} (v_0, \mathtt{out}(n_V)).(p_0, \mathtt{in}(n_V)).(p_0, \mathtt{out}(n_P)).(v_0, \mathtt{in}(n_P)) \\ (v_0, \mathtt{out}(c)).(p_0, \mathtt{in}(c)).(p_0, \mathtt{out}(\mathsf{ans}(c, m_0, m_1))).(v_0, \mathtt{in}(\mathsf{ans}(c, m_0, m_1))) \end{cases}$$

where $m_0 = \mathsf{kdf}(\mathsf{shk}(p_0,v_0), n_V, n_P)$ and $m_1 = \mathsf{shk}(p_0,v_0) \oplus m_0$.

Regarding condition (iii), we have that θ_P as defined in Example 4 and $\sigma = \{x_N \mapsto n_P\}$ satisfy our requirement. Regarding condition *(iv)*, we have that $u_1 = y_0$, and $u_2 = y_1$, and thus $C_{\mathcal{P}}$ only contains the quasi-free symbol ans.

According to our definition, when analysing a protocol $\mathcal{P}_{\mathsf{prox}}$ w.r.t. terrorist frauds you should consider all the possible semi-dishonest provers. However, for the class of distance bounding protocol we consider, we will show that we can restrict our attention to a particular dishonest prover that we define now.

Definition 8. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol as given in Definition 7. The* most general semi-dishonest prover *for $\mathcal{P}_{\mathsf{prox}}$, denoted $\mathtt{P}^*$, is the process:*

$$\big(\mathtt{block}_P.\mathtt{out}(u_1)\dots\mathtt{out}(u_k).\mathtt{in}(y_c).\mathtt{out}(u).\mathtt{block}'_P\big)\{z_0 \mapsto p_0, z_1 \mapsto v_0\}$$

where $u_1, \dots, u_p$ are the terms such that $u = C_{\mathcal{P}}[y_c, u_1, \dots, u_p]$.

Its associated frame*, denoted Φ^*, is the one obtained considering the normal execution and letting the attacker answer to the challenge relying on $C_{\mathcal{P}}$.*

The most general semi-dishonest prover will help his accomplice by sending him (before the rapid phase starts) the material he needs to perform this phase alone. For this, the most general semi-dishonest prover will send messages corresponding to the maximal subterms of u that do not contain the challenge. This will be sufficient to answer to the challenge sent by the verifier, and we will see that this is actually the strategy that leaks the least information.

Example 17. Going back to our running example, P^1_{sd} together with frame Φ^1_{sd} as described in Example 11 corresponds to the most general semi-dishonest prover.

Note that the most general semi-dishonest prover $\mathtt{P}^*$ (as given in Definition 8) is a dishonest prover. This simply means that such a process when put together with $\lfloor \mathtt{V}(v_0, p_0) \rfloor^0_{v_0}$ can be fully executed considering the topology $\mathcal{T}^0_{\mathsf{simple}}$. This is actually an easy consequence of our definition of DB protocol exploiting the fact that $\mathtt{P}^*$ and $\mathsf{P}(p_0, v_0)$ are rather similar. More interestingly, we can establish a strong relationship between the frame Φ^* (the one associated to $\mathtt{P}^*$) and a frame Φ_{sd} associated to an arbitrary semi-dishonest prover $\mathtt{P}_{\mathsf{sd}}$.

Proposition 2. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol, and $\mathtt{P}^*$ be its most general semi-dishonest prover with Φ^* its associated frame. Let exec^* be the execution witnessing the fact that $\mathtt{P}^*$ together with Φ^* is a semi-dishonest prover, i.e.:*

$$\mathsf{exec}^* : (\{\lfloor \mathtt{V}(v_0, p_0)\rfloor^0_{v_0}, \lfloor \mathtt{P}^*\rfloor^0_{p_0}\}; \emptyset; 0) \xrightarrow{\mathsf{tr}^*}_{\mathcal{T}^0_{\mathsf{simple}}} (\{\lfloor \mathsf{end}(v_0, p_0)\rfloor^{t^*_v}_{v_0}, \lfloor 0\rfloor^{t^*_p}_{p_0}\}; \Phi^*; t^*).$$

Let $\mathtt{P}_{\mathsf{sd}}$ be a semi-dishonest prover for $\mathcal{P}_{\mathsf{prox}}$ together with its frame Φ_{sd}, and exec be the execution witnessing this fact, i.e.

$$\mathsf{exec} : (\{\lfloor \mathtt{V}(v_0, p_0)\rfloor^0_{v_0}, \lfloor \mathtt{P}_{\mathsf{sd}}\rfloor^0_{p_0}\}; \emptyset; 0) \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{simple}}} (\{\lfloor \mathsf{end}(v_0, p_0)\rfloor^{t_v}_{v_0}, \lfloor 0\rfloor^{t_p}_{p_0}\}; \Phi_{\mathsf{sd}}; t).$$

We have that there exists a substitution $\sigma : \mathcal{N} \to \mathcal{T}(\Sigma_c^+, \mathcal{N} \cup \mathcal{A})$ from names freshly generated by $\mathtt{P}^*$ to constructor terms such that for any $\mathtt{out}(u)$ occurring in tr^*, there exists a recipe R such that $R\Phi_{\mathsf{sd}}\downarrow =_{\mathsf{E}} u\sigma$.

Roughly, up to some substitution σ, we know that an arbitrary dishonest prover will disclose more information than the general one. Thus, to analyse terrorist fraud resistance, it is sufficient to consider the most general semi-dishonest prover. This actually corresponds to the best strategy for the terrorist prover.

Theorem 2. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol and $\mathcal{I}_0$ be a template. Let Φ^* be the frame associated to the most general semi-dishonest prover of $\mathcal{P}_{\mathsf{prox}}$. We have that $\mathcal{P}_{\mathsf{prox}}$ is terrorist fraud resistant w.r.t. t_0-proximity if, and only if, there exists a*

topology $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{MF}}$ *and a valid initial configuration* K_0 *for* $\mathcal{P}_{\mathsf{prox}}$ *w.r.t.* $\mathcal{T}$ *and* $\Phi^* \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$ *such that:*

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathit{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

We establish this result by showing that an execution trace tr starting with $\Phi^* \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$ as an initial frame can be mimicked by an execution trace $\mathsf{tr}\sigma$ starting with the initial frame $\Phi_{\mathsf{sd}} \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$. In other words, P_{sd} is not better than P^*: the information leaked by Φ_{sd} will also allow the accomplice to authenticate again.

4.3 Main result

Applying Theorem 1 to reduce the topology, and then Theorem 2 to narrow down the number of semi-dishonest provers to consider, we get rid of the quantifications over semi-dishonest provers as well as the one regarding the topology. We now state our main reduction result.

Corollary 1. *Let* $\mathcal{P}_{\mathsf{prox}}$ *be a DB protocol and* $\mathcal{I}_0$ *be a template such that* $\mathcal{P}$ *is* $\mathcal{I}_0$*-executable. Let* P^* *be the most general semi-dishonest prover for* $\mathcal{P}$ *together with its associated frame* Φ^**. We have that* $\mathcal{P}_{\mathsf{prox}}$ *is terrorist fraud resistant w.r.t.* t_0*-proximity if, and only if, there exists a valid initial configuration* K_0 *for* $\mathcal{P}_{\mathsf{prox}}$ *w.r.t.* $\mathcal{T}^0_{\mathsf{MF}}$ *and* $\Phi^* \cup \Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0}$ *such that:*

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}^{t_0}_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi; t).$$

We will see in the next section that our definition of DB protocol is quite general and covers most existing distance bounding protocols. However, some existing protocols, like Brands & Chaum [8] and MAD [10], do not qualify for our approach. The former does not satisfy our hypothesis *(iv)* whereas the latter starts with a commit on the challenge value, preventing it to be fresh (hypothesis *(i)* is not satisfied). Despite this, since these two protocols are subject to a terrorist fraud, we could simply exhibit the corresponding semi-dishonest prover and use our methodology to establish the existence a terrorist fraud.

A reduction result allowing one to get rid of the quantification over semi-dishonest provers is also suggested in [24]. The reduction result is not formally stated. Instead, the authors provide a manual proof of resistance to terrorist fraud for a specific DBToy protocol relying on the idea of least-disclosing message. Then, the authors claim that similar proofs can be done on all the case studies they have looked at. We would like to emphasise that even if our conditions (expressed in Definition 7) are not necessarily tight, the freshness of the challenge just as the rapid phase starts is necessary to ensure the completeness of our approach. Otherwise, a best strategy for the semi-dishonest prover could be to send out a message which contains the challenge. This condition is missing in [24] and therefore their approach is *not* complete even for protocols satisfying their least-disclosing message assumption.

Protocols	MFR	TFR	Protocols	MFR	TFR
Basin's Toy Example [4]	✓	✓	Swiss-Knife [22]	✓	✓
Hancke and Kuhn [20]	✓	×	Modified Swiss-Knife [18]	✓	×
Modified Hancke and Kuhn [29]	✓	✓	Munilla *et al.* [27]	✓	×
TREAD-PKey V1 [3]	×	✓	SPADE [9]	×	✓
TREAD-PKey V1 Fixed [19]	✓	✓	SPADE Fixed [19]	✓	✓
TREAD-PKey V2 [3]	×	✓	SKI [7]	✓	✓
TREAD-PKey V2 Fixed [19]	✓	✓	PaySafe [12]	✓	×
TREAD-SKey [3]	✓	✓	NXP [21]	✓	×

Table 1: Results on our case studies (×: attack found, ✓: proved secure)

5 Case studies

Getting rid of the quantifications, we apply techniques already used in e.g. [12, 14], to leverage the verification tool ProVerif to analyse terrorist fraud on distance bounding protocols.

5.1 Analysing terrorist-fraud resistance using Proverif

Based on the technique described in [14], we reuse the syntax of phases included in Proverif to model the guarded input of a Verifier. We will consider the same transformation, while adding an extra phase at the beginning (phase 0) to enrich the knowledge of the attacker with the frame provided together with the most-general semi-dishonest prover. Then, we consider a Verifier-Test modelled using three phases (1, 2 and 3) to see whether the adversary can re-authenticate itself by impersonating the Prover or not. As in [14], we also give to the adversary the possibility to play with all the agents present in the topology if they are close enough, since they can provide useful information.

Depending on Proverif outputs, we conclude on the terrorist-fraud security of the distance-bounding protocol. Either it is not possible to reach the `end` event of the Verifier-Test, or the tool returns a trace in which the event is reachable. In the first case, the attacker can not authenticate itself to the Verifier, even with the help provided by the Prover in phase 0, meaning that the protocol is vulnerable to a terrorist-fraud attack. In the second case, we first need to ensure that the trace provided by ProVerif is a valid trace in our model. If this is the case, then the adversary can authenticate itself to the Verifier again, without further help from the Prover, meaning that the protocol is terrorist-fraud resistant. Note that, even if in theory, our approach may not allow one to conclude (in case e.g. ProVerif does not terminate or simply say cannot be proved), we never encountered this situation when performing our case studies.

5.2 Our results

We applied this methodology to different well-known distance-bounding protocols as long as they met the hypotheses needed by our approach, as mentioned

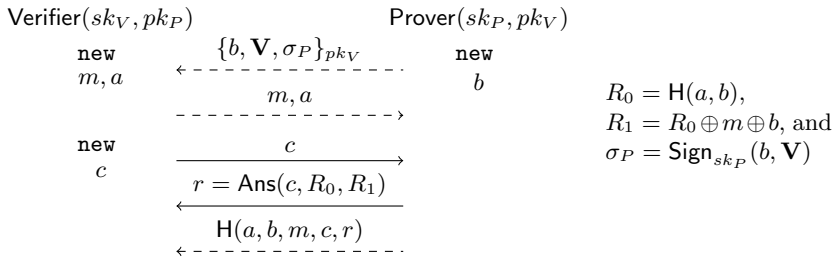


Fig. 1: Description of the SPADE (**fixed**) protocol

in Section 4. As expected, numerous existing protocols qualify and the results are shown in Table 1. All our implementation files can be found in [15]. The tool concludes in less than a minute for most of the examples, except for two protocols: SPADE and SKI, where the extensive use of the xor operator may explain this noteworthy difference. To comply with the use of the symbolic approach, we needed to replace the actual bit-sized rapid exchanges by a single round of challenge-response using one fresh nonce, as presented in the examples throughout this paper. Moreover, due to Proverif limitations, we only considered, when needed, a weak version of the xor operator.

Our results confirm existing mafia frauds against the SPADE and the TREAD (PKey version) protocols [14, 23]. Therefore we considered the fixed versions of these protocols mentioned in [19] and proved them mafia-fraud and terrorist-fraud resistant using our methodology. The fix, which consists in adding the Verifier identity in the first message sent by the Prover, is illustrated for the SPADE protocol in the Figure 1 above.

We also extended our analysis to contactless payment protocols, e.g. Paysafe and NXP. While it was not surprising that they do not offer terrorist-fraud resistance, we consider that those protocols should claim if they want to support such a security property or not. Indeed, allowing terrorist fraud could be a feature of the card, permitting its user to agree for a one-time payment to a third-party while not being physically next to it, without risking any non-expected following payment, similarly to the current virtual credit card system.

5.3 Limitations

Even if our methodology is general enough to deal with a number of examples, we had to cope with some limitations. First, coming from the tool, Proverif, as mentioned earlier, we needed to weaken the xor operator. While our methodology could consider a different tool which deals better with such an operator, like Tamarin [26], it appears that Tamarin also behaves poorly depending on the considered protocols. Indeed, we faced up with non-termination issues when we tried to apply our methodology to the Brands and Chaum protocol within Tamarin. These termination issues are also visible in the case studies performed by Mauw *et al* in [23] (using Tamarin) in which they often had to weaken the xor operator.

Second, as already discussed in 4.3, some limitations are due to the hypotheses we need on a distance-bounding protocols to conduct our formal development. We believe that we could relax our hypothesis regarding the freshness of the challenge up to a non-deductibility hypothesis to be able to apply our methodology to a protocol like MAD, but this is left to future work.

References

1. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115. ACM Press, 2001.
2. G. Avoine, M. A. Bingöl, S. Kardas, C. Lauradoux, and B. Martin. A framework for analyzing RFID distance bounding protocols. *Journal of Computer Security*, 19(2):289–317, 2011.
3. G. Avoine, X. Bultel, S. Gambs, D. Gerault, P. Lafourcade, C. Onete, and J.-M. Robert. A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol. In *Proc. 12th ACM Asia Conference on Computer and Communications Security*. ACM Press, 2017.
4. D. Basin, S. Capkun, P. Schaller, and B. Schmidt. Formal reasoning about physical properties of security protocols. *ACM Transactions on Information and System Security (TISSEC)*, 14(2):16, 2011.
5. B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proc. 14th Computer Security Foundations Workshop (CSFW'01)*, pages 82–96. IEEE Computer Society Press, 2001.
6. B. Blanchet. Modeling and verifying security protocols with the applied pi calculus and proverif. *Foundations and Trends in Privacy and Security*, 1(1-2):1–135, 2016.
7. I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Secure and lightweight distance-bounding. In *International Workshop on Lightweight Cryptography for Security and Privacy*, pages 97–113. Springer, 2013.
8. S. Brands and D. Chaum. Distance-bounding protocols. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 344–359. Springer, 1993.
9. X. Bultel, S. Gambs, D. Gerault, P. Lafourcade, C. Onete, and J. Robert. A prover-anonymous and terrorist-fraud resistant distance-bounding protocol. In *Proc. 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, (WISEC'16)*, pages 121–133. ACM Press, 2016.
10. S. Čapkun, L. Buttyán, and J.-P. Hubaux. Sector: secure tracking of node encounters in multi-hop wireless networks. In *Proc. 1st ACM workshop on Security of ad hoc and sensor networks*, pages 21–32. ACM, 2003.
11. T. Chothia, J. de Ruiter, and B. Smyth. Modelling and analysis of a hierarchy of distance bounding attacks. In *Proc. 27th USENIX Security Symposium, USENIX Security 2018*, 2018.
12. T. Chothia, F. D. Garcia, J. de Ruiter, J. van den Breekel, and M. Thompson. Relay cost bounding for contactless EMV payments. In *Proc. 19th International Conference on Financial Cryptography and Data Security (FC'15)*, volume 8975 of *LNCS*. Springer, 2015.
13. A. Debant and S. Delaune. Symbolic verification of distance bounding protocols. In *Proc. 8th International Conference on Principles of Security and Trust (POST'19)*, LNCS. Springer, 2019.

14. A. Debant, S. Delaune, and C. Wiedling. A symbolic framework to analyse physical proximity in security protocols. In *Proc. 38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, (FSTTCS'18)*, volume 122 of *LIPIcs*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.
15. A. Debant, S. Delaune, and C. Wiedling. Symbolic Analysis of Terrorist Fraud Resistance. Research report, Univ Rennes, CNRS, IRISA, France, July 2019.
16. D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on information theory*, 29(2):198–208, 1983.
17. U. Dürholz, M. Fischlin, M. Kasper, and C. Onete. A formal approach to distance-bounding RFID protocols. In *Proc. 14th International Conference on Information Security (ISC'11)*, volume 7001 of *LNCS*. Springer, 2011.
18. M. Fischlin and C. Onete. Terrorism in distance bounding: Modeling terrorist-fraud resistance. In *Proc. 11th International Conference on Applied Cryptography and Network Security (ACNS'13)*, volume 7954 of *LNCS*. Springer, 2013.
19. D. Gerault. *Security Analysis of Contactless Communication Protocols*. PhD thesis, Université Clermont Auvergne, 2018.
20. G. P. Hancke and M. G. Kuhn. An RFID distance bounding protocol. In *Proc. 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 67–73. IEEE, 2005.
21. P. Janssens. Proximity check for communication devices, Oct. 31 2017. US Patent 9,805,228.
22. C. H. Kim, G. Avoine, F. Koeune, F. Standaert, and O. Pereira. The Swiss-Knife RFID distance bounding protocol. In *Proc. 11th International Conference on Information Security and Cryptology (ICISC'08)*, volume 5461 of *LNCS*. Springer, 2008.
23. S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *Proc. 39th IEEE Symposium on Security and Privacy (S&P'18)*, pages 152–169, 2018.
24. S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua. Post-Collusion Security and Distance Bounding. In *(To appear in) Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2019.
25. C. Meadows, R. Poovendran, D. Pavlovic, L. Chang, and P. 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. Springer, 2007.
26. S. Meier, B. Schmidt, C. Cremers, and D. Basin. The Tamarin Prover for the Symbolic Analysis of Security Protocols. In *Proc. 25th International Conference on Computer Aided Verification (CAV'13)*, volume 8044 of *LNCS*, pages 696–701. Springer, 2013.
27. J. Munilla and A. 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.
28. V. Nigam, C. Talcott, and A. A. Urquiza. Towards the automated verification of cyber-physical security protocols: Bounding the number of timed intruders. In *Proc. 21st European Symposium on Research in Computer Security (ESORICS'16)*, pages 450–470. Springer, 2016.
29. S. Vaudenay. On modeling terrorist frauds - addressing collusion in distance bounding protocols. In *Proc. 7th International Conference on Provable Security (ProvSec'13)*, volume 8209 of *LNCS*, pages 1–20. Springer, 2013.
30. S. Vaudenay, I. Boureanu, A. Mitrokotsa, et al. Practical & provably secure distance-bounding. In *Proc. 16th Information Security Conference (ISC'13)*, 2013.