Symbolic analysis of terrorist fraud resistance

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

[1] Univ Rennes, CNRS, IRISA, France
[2] DGA MI, Bruz, France

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 terminal of payment and a credit card. As a result, the terminal will consider that the credit card is close to it and will accept the payment, whereas the card is far away, and perhaps 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 the light. Since they have been introduced by Brands and Chaum in 1993 [9], many protocols have been designed and analysed against various threats. Clearly, 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 relay attacks where typically an attacker abuses a far away prover to pass the protocol (so-called *mafia fraud*). 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. The protocol is resistant to a terrorist fraud if this help is actually reusable meaning that the attacker 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 [6, 7], Tamarin [25]. 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. [24, 5, 27]), some recent works have proposed to incorporate new features in existing symbolic models [23, 15, 12, 14], 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 [15]. This calculus shares some similarities with the applied pi calculus [2, 7], 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, 15, 14]. 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 [15, 12], 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 [1].

Related works. Several attempts have been made in the computational model to formalise terrorist fraud, e.g. [17, 3, 18, 29]. Avoine *et al.* [3] 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, 29], as well as protocols supposed to achieve this level of security, e.g. [22, 10, 4]. In contrast, the only definition we are aware of in the symbolic model is the one proposed by Chothia et al in [12]. However, such a definition falls short when modelling behaviours of terrorist provers (see Section 3).

2 Model for distance bounding protocols

In this section, we introduce the process calculus we rely on to describe distance bounding protocols [15]. It shares some similarities with the applied pi calculus used e.g. by the ProVerif verification tool [7].

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 test, 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 [7], 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]

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

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 [7], 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: $x_0 \mapsto m_0; x_1 \mapsto k \oplus m_0; x_{\mathsf{rep}} \mapsto \mathsf{ans}(c, m_0, k \oplus m_0);\ x_{\mathsf{ok}} \mapsto \mathsf{ok}$.

2.2 Protocols

Protocols are modeled 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 *parametrised 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 parametrised 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 [28]). It relies on the use of a keyed public pseudo-random function (modeled as a free function symbol here) and the exclusive-or operator.

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

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:

```
V(z_0, z_1) :=
  new n_V.out(n_V).in(x_N).
  reset.new c.out(c).in^{<2×t_0}(x_rep).
  let x_0 = kdf(shk(z_1, z_0), n_V, x_N) in
  let x_1 = shk(z_1, z_0) ⊕ x_0 in
  let x_ok = eq(x_rep, ans(c, x_0, x_1)) in
  end(z_0, z_1)
```

```
P(z_0, z_1) :=
  new n_P.in(y_N).out(n_P).
  let y_0 = kdf(shk(z_0, z_1), y_N, n_P) in
  let y_1 = shk(z_0, z_1) ⊕ y_0 in
  in(y_c).
  out(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, 15, 12, 14], the rapid phase is abstracted by a single challenge/response exchange, and operations performed at the bit level are abstracted too.

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

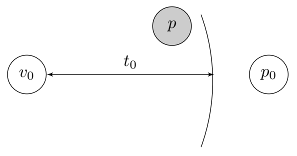
Configuration. The semantics of our processes is given through a transition system defined over configurations. Given a topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$, a *configuration* K over $\mathcal{T}_0$ is a tuple $(\mathcal{P}; \Phi; t)$, where:

- $\mathcal{P}$ 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 the semantics of our calculus as defined in [15]. A full description of the semantics is given in Appendix for sake of completeness. Below, we only describe few rules to give a flavour of it.

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

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

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

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 $b \in \mathcal{A}_0 \smallsetminus \mathcal{M}_0$ 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 Tim rule allows time to elapse, meaning the the global clock as well as the local clocks of each agent 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 Out rule is used to output a message which is immediately added into the frame. The In rule is rather standard despite some additional side conditions which allow 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. The rules Let, Rst, and New allowing one to take care of the remaining actions are quite straightforward and given in Appendix.

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:

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

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 (1st 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 [15] 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.

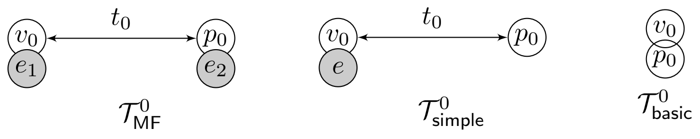
Definition 2. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathit{Loc}_0, v_0, p_0)$ be a topology, and Φ_0 be an initial frame. $K = (\mathcal{P}; \Phi; t)$ is a* valid initial configuration *for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}_0$ 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 $P' = \mathsf{V}(a', b')$ or $P' = \mathsf{P}(a', b')$ 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 [15]. 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:


t_0
v_0
p_0
e_1
e_2
$\mathcal{T}^0_{\mathsf{MF}}$
t_0
v_0
p_0
e
$\mathcal{T}^0_{\mathsf{simple}}$
v_0
p_0
$\mathcal{T}^0_{\mathsf{basic}}$

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

Example 10. Continuing our running example, and considering the topology $\mathcal{T}_{\mathsf{MF}}^0 = (\mathcal{A}_{\mathsf{MF}}^0, \mathcal{M}_{\mathsf{MF}}^0, \mathsf{Loc}_{\mathsf{MF}}^0, v_0, p_0)$ (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 $\mathsf{Knows}_{\mathcal{I}_0}(e_i, \mathcal{A}_{\mathsf{MF}}^0)$ will contain all the symmetric keys this malicious agent shares with other agents.

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

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 $\Phi_{\mathcal{I}_0}^{\mathcal{T}}$ such that:*

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

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

3.2 Terrorist fraud

Modelling terrorist fraud is tricky. We have to consider a semi-dishonest prover who colludes with the attacker with the aim of letting him authenticating 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 $\mathcal{T}_{\mathsf{simple}}^0$ 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:*

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

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 ($k = \mathsf{shk}(v_0, p_0)$, $m_0 = \mathsf{kdf}(k, n_V^0, n_P^0)$, and $m_1 = k \oplus m_0$):

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

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 $\Phi_{\mathcal{I}_0}^{\mathcal{T}} \cup \Phi_{\mathsf{sd}}$ such that:*

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

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):

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

This execution witnesses the fact that the dishonest prover P_{sd}^2 together with frame Φ_{sd}^2 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 re-authenticate itself again. Distance bounding protocols designed to achieve terrorist fraud resistance aim also to resist against mafia fraud. So, degrading a protocol to make it vulnerable to a mafia fraud is not an interesting option.

Up to our knowledge, the only existing definition of terrorist fraud resistance in the symbolic setting is the one proposed by Chothia *et al* in [12]. 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. 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 (see Examples in Appendix).

The main advantage of the definition of terrorist fraud proposed by [12] 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 [12] is quite debatable, it is fixed, and can therefore be given in input to the verification tool. Our definition of terrorist fraud resistance which quantifies over all the possible semi-dishonest provers is more complex, and actually no existing verification tool is able to handle this quantification. In Section 4, we will show how to get rid of this quantification as well as the one regarding the topology.

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 prover.

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 [15] regarding mafia and distance hijacking frauds. The only issue 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 13. 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 $i \in \{1, \dots, n\} \smallsetminus \{i_0, i_0 + 1\}$.

(iii) *Let $\mathcal{U} = \{x = u \mid$ "$\mathtt{let}\ x = u\ \mathtt{in}$ " 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 interfere, 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 seems 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 $u_1, \ldots, u_n$ in advance to his accomplice, and let him to compute (as indicated by C) the answer to the challenge from $u_1, \ldots, u_n$ 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 14. 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 P^*, is the process:*

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

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

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 less information.

Example 15. 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 $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathit{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{MF}}$ 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:*

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

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 $\mathtt{P}^*$: the information leaked by Φ_{sd} will also allow the accomplice to authenticate again.

5 Case studies

Using the results to reduce the topology (Theorem 1) and narrow down the number of semi-dishonest provers to consider (Theorem 2), we got rid of the quantifications over semi-dishonest provers as well as the one regarding the topology.

Protocols	MFR	TFR	Protocols	MFR	TFR
Basin's Toy Example [5]	✓	✓	TREAD-PKey V1 [4]	×	✓
Hancke and Kuhn [20]	✓	×	TREAD-PKey V1 Fixed [19]	✓	✓
Modified Hancke and Kuhn [28]	✓	✓	TREAD-PKey V2 [4]	×	✓
Swiss-Knife [22]	✓	✓	TREAD-PKey V2 Fixed [19]	✓	✓
Modified Swiss-Knife [18]	✓	×	TREAD-SKey [4]	✓	✓
Munilla *et al.* [26]	✓	×	SKI [8]	✓	✓
SPADE [10]	×	✓	PaySafe [13]	✓	×
SPADE Fixed [19]	✓	✓	NXP [21]	✓	×

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

This result is formally stated in Appendix. Now, applying techniques already used in e.g. [13, 15], we show how to leverage the verification tool ProVerif to analyse terrorist fraud on distance bounding protocols.

5.1 Analyzing terrorist-fraud resistance using Proverif

Based on the technique described in [15], 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 modeled using three phases (1, 2 and 3) to see whether the adversary can re-authenticate itself by impersonating the Prover or not. As in [15], 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 in Section 4. As expected, numerous existing protocols qualify and the results are shown in Table 1. All our implementation files can be found at [1]. The tool

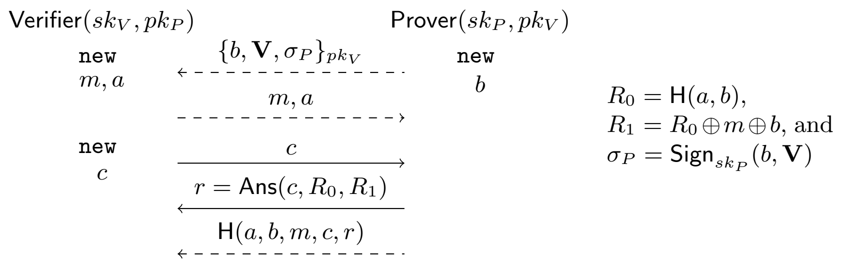

Verifier(sk_V, pk_P)
Prover(sk_P, pk_V)
new m, a
$\{b, \mathbf{V}, \sigma_P\}_{pk_V}$
new b
m, a
new c
c
$r = \mathsf{Ans}(c, R_0, R_1)$
$\mathsf{H}(a, b, m, c, r)$
$R_0 = \mathsf{H}(a, b)$,
$R_1 = R_0 \oplus m \oplus b$, and
$\sigma_P = \mathsf{Sign}_{sk_P}(b, \mathbf{V})$

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

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.

As shown in Table 1, SPADE and TREAD (PK version) protocols are subject to a mafia fraud, therefore we considered fixed versions of these protocols proved to be 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 [25], it appears that Tamarin also behaves poorly depending on the considered protocols. Second, some limitations are due to the hypotheses we need on a distance-bounding protocols to conduct our formal development. Some existing protocols, like Brands & Chaum [9] and MAD [11], do not qualify for our approach. The former do 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). We believe that we could relax the freshness of the challenge hypothesis up to a non-deductibility hypothesis to be able to apply our methodology to this kind of protocols, but this is left to future work.

References

1. http://people.irisa.fr/Alexandre.Debant/proverif-tfr.html.
2. 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.
3. 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.
4. 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. S. Brands and D. Chaum. Distance-bounding protocols. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 344–359. Springer, 1993.
10. 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.
11. 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.
12. 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.
13. 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.
14. 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.
15. 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.
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. 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.

25. 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.

26. 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.

27. 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.

28. 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.

29. S. Vaudenay, I. Boureanu, A. Mitrokotsa, et al. Practical & provably secure distance-bounding. In *Proc. 16th Information Security Conference (ISC'13)*, 2013.

A Main appendix

A.1 Semantics

$$\begin{array}{lll}
\textsc{Tim} & (\mathcal{P};\Phi;t) \longrightarrow_{\mathcal{T}_0} (\mathsf{Shift}(\mathcal{P},\delta);\Phi;t+\delta) & \text{with } \delta \geq 0 \\
\textsc{Out} & (\lfloor \mathtt{out}(u).P \rfloor_a^{t_a}) \uplus \mathcal{P};\Phi;t) \xrightarrow{a,\mathtt{out}(u)}_{\mathcal{T}_0} (\lfloor P \rfloor_a^{t_a} \uplus \mathcal{P};\Phi \uplus \{\mathsf{w} \xrightarrow{a,t} u\};t) & \text{with } \mathsf{w} \in \mathcal{W} \text{ fresh} \\
\textsc{Let} & (\lfloor \mathtt{let}\ x = u\ \mathtt{in}\ P \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P\{x \mapsto u{\downarrow}\} \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) & \text{when } u{\downarrow} \in \mathcal{T}(\Sigma_c^+, \mathcal{N} \uplus \mathcal{A}) \\
\textsc{New} & (\lfloor \mathtt{new}\ n.P \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P\{n \mapsto n'\} \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) & \text{with } n' \in \mathcal{N} \text{ fresh} \\
\textsc{Rst} & (\lfloor \mathtt{reset}.P \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) \xrightarrow{a,\tau}_{\mathcal{T}_0} (\lfloor P \rfloor_a^{0} \uplus \mathcal{P};\Phi;t) & \\
\textsc{In} & (\lfloor \mathtt{in}^\star(x).P \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) \xrightarrow{a,\mathtt{in}^\star(u)}_{\mathcal{T}_0} (\lfloor P\{x \mapsto u\} \rfloor_a^{t_a} \uplus \mathcal{P};\Phi;t) &
\end{array}$$

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

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

The Tim rule allows time to elapse, meaning the the global clock as well as the local clocks of each agent 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 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. let $x = \mathsf{dec}(y,k)$ 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) names, and the Rst rule allows an agent to reset the local clock of the process. The In rule is rather standard despite some additional side conditions which allow 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.

A.2 Comparison with [12]

In this section we develop two examples illustrating the main differences between the approach proposed by Chothia *et al* in [12] and ours. In the first example, we highlight that it may be impossible to write a finite process corresponding to their notion of terrorist prover. In the second example, we show that a syntactic definition (as the one they proposed) does not seem suitable.

Example 16. In this example, we assume for instance that the protocol relies on a hash function h and 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: $\mathtt{in}(x).\mathtt{out}(\mathsf{h}(\langle k, x\rangle))$. In the same spirit, we could argue that $\mathtt{in}(x).\mathtt{out}(\mathsf{h}(\langle x, k\rangle))$ is also useful, and perhaps also $\mathtt{in}(x_1).\mathtt{in}(x_2).\mathtt{out}(\mathsf{h}(\langle x_1, \langle k, x_2\rangle\rangle))$, 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.

Example 17. In [12], Chothia *et al* do not precise in which measure the term algebra they consider has an impact on the definition of the terrorist prover. Going back to our running example, a terrorist prover will be allowed to perform the exclusive or operation with the secret k, i.e. the oracle $\mathtt{in}(x).\mathtt{out}(x \oplus k)$ will be part of the terrorist prover. This oracle actually reveals the k. Indeed, an attacker learning $\mathtt{c} \oplus k$ for some public constant $\mathtt{c}$ will be able to retrieve k. This oracle is too strong and we think that it should not be part of the terrorist prover.

An even more complex situation occurs when considering some specific primitives. For instance, assume that we have a rewriting rule of the form

$$\mathtt{g}(\mathtt{f}_1(x, y), \mathtt{f}_2(x, y)) = y$$

where $\mathtt{f}_1$ and $\mathtt{f}_2$ are two constructor symbols whereas $\mathtt{g}$ is a destructor symbol. Following the idea developed in [12], 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.

A.3 Main result combining Theorems 1 and 2

As a consequence of our two reduction results, we have the following corollary:

Corollary 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol and $\mathcal{I}_0$ be a template such that $\mathcal{P}_{\mathsf{prox}}$ is $\mathcal{I}_0$-executable. Let $\mathtt{P}^*$ be the most general semi-dishonest prover for $\mathcal{P}_{\mathsf{prox}}$ 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 $\Phi^* \cup \Phi_{\mathcal{I}_0}^{\mathcal{T}^0_{\mathsf{MF}}}$ such that:*

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

B Proofs of our results

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).*

Proof. If $\mathcal{P}_{\mathsf{prox}}$ admits a mafia fraud w.r.t. t_0-proximity then 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 $K = (\mathcal{P}_0; \Phi_0; 0)$ for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^{\mathcal{T}}_{\mathcal{I}_0}$ such that:

$$K \rightarrow^*_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi_0 \cup \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0$$

for some multiset of extended processes $\mathcal{P}$, frame Φ and times $t_v, t \in \mathbb{R}_+$.

For all Φ_{sd} together with a frame Φ_{sd} a semi-dishonest prover for $\mathcal{P}_{\mathsf{prox}}$, we have that $K' = (\mathcal{P}_0; \Phi_0 \cup \Phi_{\mathsf{sd}}; 0)$ is a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}$. The same execution applies since the initial knowledge of the attacker has just been increased.

We can thus conclude that $\mathcal{P}_{\mathsf{prox}}$ is terrorist fraud resistant. □

B.1 Proofs of Section 4.1

This reduction result is very close to the one stated and proved in [15] for mafia fraud. However, we have to take care of the extra messages added into the frame that may contain some agent names (other than v_0, p_0, e_1, and e_2). Actually, the transformation applied to reduce the topology will modify these names. Thus, to make sure that the reduction result given in [15] applies, we will first show that names other than v_0, p_0, e_1, and e_2 are not mandatory in the initial frame.

Lemma 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be an $\mathcal{I}_0$-executable protocol w.r.t. some template $\mathcal{I}_0$. Moreover, we assume that there exists a valid initial configuration 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 $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t)$ for any semi-dishonest prover P_{sd} with frame Φ_{sd} such that $names(\Phi_{\mathsf{sd}}) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$.*

Let P'_{sd} be a semi-dishonest prover with frame Φ'_{sd}. We have that 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

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

Proof. We consider P'_{sd} a semi-dishonest prover with frame Φ'_{sd}. If $names(\Phi_{\mathsf{sd}}) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$ then the result is immediate. Otherwise, let us consider a bijective renaming σ that allows one to rename agent names occurring in $(names(\Phi'_{\mathsf{sd}}) \cap \mathcal{A}) \smallsetminus \{v_0, p_0, e_1, e_2\}$ to fresh constants in Σ_0. By fresh, we mean that these constants do not occur in $\mathcal{P}_{\mathsf{prox}}$ and $\mathcal{I}_0$. We have that $P'_{\mathsf{sd}}\sigma$ is a semi-dishonest prover with frame $\Phi'_{\mathsf{sd}}\sigma$.

By construction we have that $names(\Phi'_{\mathsf{sd}}\sigma) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$ and thus our hypothesis applies. We have that there exists a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}^0_{\mathsf{MF}}$ and $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}\sigma$ such that

$$\mathtt{exec} = K \xrightarrow{\mathsf{tr}}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t).$$

Applying the renaming σ^{-1} along $\mathtt{exec}$, we obtain that:

$$K\sigma^{-1} \xrightarrow{\mathsf{tr}\sigma^{-1}}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}\sigma^{-1}; \Phi\sigma^{-1}; t)$$

Moreover, we know that K is a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}^0_{\mathsf{MF}}$ and $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}\sigma$ and thus $K = (\mathcal{Q}; \Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}\sigma; 0)$ where $\mathcal{Q}$ is a multiset of extended processes such that for all $\lfloor P' \rfloor^0_a \in \mathcal{Q}$, we have that $a \in \{v_0, p_0, e_1, e_2\}$ and there exists $b \in \{v_0, p_0, e_1, e_2\}$ such that $P' = \mathtt{P}(a, b)$ or $P' = \mathtt{V}(a, b)$. Since σ only replace agent names by fresh constants, we can conclude that $\mathcal{Q}\sigma^{-1} = \mathcal{Q}$ and $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0}\sigma^{-1} = \Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0}$. Finally we have:

$$(\mathcal{P}; \Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}\sigma\sigma^{-1}; 0) \xrightarrow{\mathsf{tr}\sigma^{-1}}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}\sigma^{-1}; \Phi\sigma^{-1}; 0).$$

with $(\mathcal{P}; \Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}\sigma\sigma^{-1}; 0)$ a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi'_{\mathsf{sd}}$. This concludes the proof. □

Then, assuming that Φ_{sd} does not contain agent names other than v_0, p_0, e_1, and e_2, as a corollary of the result stated and proved in [15], we have the following lemma.

Lemma 2. *Let $\mathcal{P}_{\mathsf{prox}}$ be an $\mathcal{I}_0$-executable protocol w.r.t. some template $\mathcal{I}_0$. Let P_{sd} be a semi-dishonest prover with frame Φ_{sd} such that names$(\Phi_{\mathsf{sd}}) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$. Let $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$ be a topology and K be a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}$ such that*

$$K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathit{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

We have that there exists K' 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

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

Proof. The proof is the same as the one presented in [15] and proceeds as follows:

(1) The exact same trace can be executed in a topology in which all the agents but v_0 and p_0 are considered dishonest;
(2) Thanks to the $\mathcal{I}_0$-executability of the protocol, all the processes executed by dishonest agents are removed from the initial configuration and the event $\mathsf{end}(v_0, p_0)$ is still reachable;
(3) Starting with a simpler configuration, the previous trace can still be executed in the simple topology $\mathcal{T}^0_{\mathsf{MF}}$;
(4) The last step consists in reducing the initial frame mapping any agent name $a \notin \{v_0, p_0, e_1, e_2\}$ to e_1. The resulting frame corresponds to $\Phi^{\mathcal{T}^0_{\mathsf{MF}}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}\rho$ where ρ replaces agent names other than v_0, p_0, e_2 by e_1. To finish our proof we note that $\Phi_{\mathsf{sd}}\rho = \Phi_{\mathsf{sd}}$ because *names*$(\Phi_{\mathsf{sd}}) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$ by hypothesis. □

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:*

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

Proof. The $\Leftarrow$ implication is trivial. It is indeed sufficient to choose the topology $\mathcal{T}^0_{\mathsf{MF}}$ to conclude. Regarding the $\Rightarrow$ implication, consider first a semi-dishonest prover P_{sd} with frame Φ_{sd} such that $names(\Phi_{\mathsf{sd}}) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$, and a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$ and $\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}$ such that:

$$K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Applying Lemma 2, we know 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: $K' \xrightarrow{\mathsf{tr}'}_{\mathcal{T}^0_{\mathsf{MF}}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'_v}_{v_0} \uplus \mathcal{P}'; \Phi'; t')$.

This allows us to conclude. Now, Lemma 1 allows us to conclude that the result holds for any semi-dishonest prover P_{sd} (even those that rely on agent names outside $\{v_0, p_0, e_1, e_2\}$). □

B.2 Proofs of Section 4.2

Lemma 3. *Let* $\mathcal{P}_{\mathsf{prox}}$ *be a DB protocol. The most general semi-dishonest process* $\mathtt{P}^*$ *together with its associated frame* Φ^* *is a semi-dishonest prover. Moreover we can assume that the trace* tr^* *witnessing this fact is such that:*

$$\begin{cases} (a_1, \mathtt{out}(m_1)).(b_1, \mathtt{in}(m_1)) \ldots (a_{i_0-1}, \mathtt{out}(m_{i_0-1})).(b_{i_0-1}, \mathtt{in}(m_{i_0-1})). \\ (p_0, \mathtt{out}(m^1_{i_0+1})). \ldots .(p_0, \mathtt{out}(m^k_{i_0+1})). \\ (v_0, \mathtt{out}(m_{i_0})).(v_0, \mathtt{in}^{<2\times t_0}(m_{i_0+1})) \\ (p_0, \mathtt{in}(m_{i_0})).(p_0, \mathtt{out}(m_{i_0+1})). \\ (a_{i_0+2}, \mathtt{out}(m_{i_0+2}).(b_{i_0+2}, \mathtt{in}(m_{i_0+2})) \ldots (a_n, \mathtt{out}(m_n).(b_n, \mathtt{in}(m_n)) \end{cases}$$

where:

- $\{a_i, b_i\} = \{v_0, p_0\}$ *for any* $i \in \{1, \ldots, n\} \smallsetminus \{i_0, i_0+1\}$*;*
- $m_{i_0+1} = C_{\mathcal{P}}[m_{i_0}, m^1_{i_0+1}, \ldots, m^p_{i_0+1}]$*;*
- $(x_1, \ldots, x_k)\theta_{\mathcal{P}}\!\downarrow\!\sigma = m_{i_1}, \ldots, m_{i_k}$ *where* $x_1, \ldots, x_k$ *are the variables occurring in input in the role* $\mathtt{V}(v_0, p_0)$*, and* $i_1, \ldots, i_k$ *are the indices among* $1, \ldots, n$ *corresponding to input performed by* v_0*, and* σ *a bijective renaming from variables to names freshly generated by* $\mathtt{P}^*$ *when executing* tr^**.*

Proof. This proof strongly relies on Definition 7. First, remark that tr^* corresponds to the trace tr in which the extra outputs of $\mathtt{P}^*$ are executed just before the ${i_0}^{\text{th}}$ communication action i.e. the output of the challenge; and the answer to the challenge received by v_0 is anticipated. We show that this sequence of actions tr^* is an execution w.r.t. our semantics:

- 1st line of actions: The actions can be executed following our semantics applying a Tim rule with a delay $\delta = t_0$ before each input. Indeed this delay enables the agent b_i to receive the message m_i sent by a_i. Moreover, since there is no guarded input the In rule always applies.

– 2nd line of actions: It only contains outputs and thus can trivially be executed.
– 3rd line of actions: Before executing the output, we apply a TIM rule to let available all the previous messages (including $m^1_{i_0+1}, \dots, m^k_{i_0+1}s$) for the malicious agent e. Since $C_{\mathcal{P}}$ only contains public symbols of functions (otherwise there is a contradiction with item (iv) of Definition 7), we have that $R = C_{\mathcal{P}}[\mathsf{w}_{i_0}, \mathsf{w}^1_{i_0+1}, \dots, \mathsf{w}^k_{i_0+1}]$ where w_{i_0} is the frame variable binding m_{i_0} and $\mathsf{w}^j_{i_0+1}$, $(1 \leq j \leq k)$ is the frame variable binding $m^j_{i_0+1}$, is a recipe deducing m_{i_0+1}. Finally the guarded input can be executed because w.l.o.g. we may assume that the reset action has been made right before the output of m_{i_0}.
– 4th and 5th lines of actions: These actions can be executed for the same reason as the first line applying a TIM rule with a delay $\delta = t_0$ before each input. □

We denote by $st(t)$ the set of syntactic subterm of a term t.

Lemma 4. *Let t_0 be a term such that $t_0{\downarrow} =_{\mathsf{E}} \mathsf{f}(u_1, \dots, u_k)$ with f a quasi-free function symbol. We have that there exist $u'_1, \dots, u'_k$ such that $\mathsf{f}(u'_1, \dots, u'_k) \in st(t_0)$ and $u'_i{\downarrow} =_{\mathsf{E}} u_i$ for any $i \in \{1, \dots, k\}$.*

Proof. Let t_0 be a term, and $t_0{\downarrow}$ its normal form. Therefore, we have that $t_0 \to t_1 \to t_2 \dots \to t_n = t_0{\downarrow}$. We prove the result by induction on the length n of this derivation.

Base case: $n = 0$. We have that $t_0{\downarrow} = t_0$, and thus $t_0 =_{\mathsf{E}} \mathsf{f}(u_1, \dots, u_k)$. Since our theory is non-trivial, and since f does not occur in equations in E, we have that there exists $u'_1, \dots, u'_k$ such that $\mathsf{f}(u'_1, \dots, u'_k) \in st(t_0)$ and $u'_i =_{\mathsf{E}} u_i$ for any $i \in \{1, \dots, k\}$. Note that $u'_1, \dots, u'_k$ are in normal form since t_0 is in normal form, and thus the result holds.

Induction step. In such a case, applying our induction hypothesis, we know that there exist $u'_1, \dots, u'_k$ such that $\mathsf{f}(u'_1, \dots, u'_k) \in st(t_1)$ and $u'_i{\downarrow} =_{\mathsf{E}} u_i$ for any $i \in \{1, \dots, k\}$. We denote $\mathsf{g}(v'_1, \dots, v'_\ell) \to v'$ the rewrite rule applied at position p to rewrite t_0 in t_1. We have that there exists a substitution θ such that $t_0|_p =_{\mathsf{E}} g(v'_1, \dots, v'_k)\theta$ and $t_1 = t_0[v'\theta]_p$. We have that $\mathsf{f}(u'_1, \dots, u'_n) \in st(t_1)$, and we distinguish two cases depending on the position p_{f} at which this subterm occurs in $t_1 = t_0[v\theta]_p$:

1. p_{f} is a position in $t_0[\_]$. In such a case, we have that either $\mathsf{f}(u'_1, \dots, u'_n) \in st(t_0)$ (in case p_{f} is not a prefix of p); or $t_0|_{p_{\mathsf{f}}} = \mathsf{f}(u''_1, \dots, u''_n)$ for some $u''_1, \dots, u''_n$, and we have that $t_0|_{p_{\mathsf{f}}} \to \mathsf{f}(u'_1, \dots, u'_n)$ with $u''_i = u'_i$ or $u''_i \to u'_i$. Therefore, we have that there exist $u''_1, \dots, u''_n$ such that $\mathsf{f}(u'_1, \dots, u'_n) \in st(t_0)$ and $u''_i{\downarrow} =_{\mathsf{E}} u_i$ for any $i \in \{1, \dots, k\}$.
2. p_{f} is a position below p, i.e. p is a prefix of p_{f}. In such a case, since f does not occur in v' (by definition of quasi-free), we have that $\mathsf{f}(u'_1, \dots, u'_k) \in st(x\theta)$ for some $x \in vars(v') \subseteq vars(v'_1, \dots, v'_\ell)$. Therefore, we have that there exists $t' =_{\mathsf{E}} t_0|_p$ such that $\mathsf{f}(u'_1, \dots, u'_k) \in st(t')$. Since we only consider non-trivial

theory, and since f does not occur in equations in E, we have that there exists $u''_1, \ldots, u''_k$ such that $\mathsf{f}(u''_1, \ldots, u''_k) \in st(t_0|_p)$ and $u''_i =_\mathsf{E} u'_i$ for any $i \in \{1, \ldots, k\}$. Note that $u''_1, \ldots, u''_k$ are in normal form since any subterm of $t_0|_p$ is in normal form. Therefore, we have that there exist $u''_1, \ldots, u''_n$ such that $\mathsf{f}(u''_1, \ldots, u''_n) \in st(t_0)$ and $u''_i\!\downarrow =_\mathsf{E} u_i$ for any $i \in \{1, \ldots, k\}$.

This concludes the proof. □

Lemma 5. *Let Φ be a frame and $c \in \mathcal{N}$ such that $c \notin st(img(\Phi))$, and $\Phi^+ = \Phi \cup \{\mathsf{w}_c \xrightarrow{v_0, t_0} c\}$. Let R be a recipe such that $R\Phi^+\!\downarrow =_\mathsf{E} u$. Let C be a context of minimal size made of quasi-free public function symbols such that $u = C[c, u_1, \ldots, u_p]$ for some $u_1, \ldots, u_p$ and c does not occur in $st(\{u_1, \ldots, u_p\})$. For any $i \in \{1, \ldots, p\}$, we have that there exists R_i such that $R_i\Phi^+\!\downarrow =_\mathsf{E} u_i$.*

Proof. We prove this result by structural induction on the context C.

Base case: C is empty. In such a case, we have that either $p = 0$; or $p = 1$ with $u_= u$. In both case, the result trivially holds.

Inductive case: $C = \mathsf{f}(C_1, \ldots, C_k)$. In such a case, by minimality of C, we know that c occurs in u. We have that $u = \mathsf{f}(u'_1, \ldots, u'_k)$ and for all $i \in \{1, \ldots, k\}$ we note $\{u^i_1, \ldots, u^i_{p_i}\} \subseteq \{u_1, \ldots, u_p\}$ the set of terms involved in the sub-context C_i i.e. $C_i[u^i_1, \ldots, u^i_{p_i}] = u'_i$. Note that $\bigcup_{1 \leq i \leq k} \{u^i_1, \ldots, u^i_{p_i}\} = \{u_1, \ldots, u_p\}$.

Applying Lemma 4 on $R\Phi^+$, we deduce that there exist $v_1, \ldots, v_k$ such that $\mathsf{f}(v_1, \ldots, v_k) \in st(R\Phi^+)$ and $v_i\!\downarrow =_\mathsf{E} u'_i$ for any $i \in \{1, \ldots, k\}$. Now, since c occurs in u we have that there exists $i_0 \in \{1, \ldots, k\}$ such that $u^{i_0}_1 = c$. Therefore we have that c occurs in $v_{i_0}\!\downarrow$ because C_{i_0} only contains quasi-free function symbols (i.e. function symbols which do not occur in E). By consequence we have that $\mathsf{f}(v_1, \ldots, v_k) \notin img(\Phi^+)$ and thus there exists $R_1, \ldots, R_k$ such that $\mathsf{f}(R_1, \ldots, R_k) \in st(R)$ with $R_i\Phi^+\!\downarrow =_\mathsf{E} u'_i$ for any $1 \leq i \leq k$.

Our induction hypothesis applies for any $i \in \{1, \ldots, k\}$ and we obtain that for all $v \in \{u^i_1, \ldots, u^i_{p_i}\}$, there exists a recipe R_v such that $R_v\Phi^+\!\downarrow =_E v$. Considering the previous remark stating that $\bigcup_{1 \leq i \leq k} \{u^i_1, \ldots, u^i_{p_i}\} = \{u_1, \ldots, u_p\}$, this allows us conclude the proof. □

Proposition 3. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol, and P^* be its most general semi-dishonest prover with Φ^* its associated frame. Let exec^* be the execution witnessing the fact that P^* together with Φ^* is a semi-dishonest prover (as given in Lemma 3). We have that exec^* is as follows:*

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

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

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

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 P^* to constructor terms such that

i if $(v_0, \mathtt{in}(u)) \in \mathsf{tr}^*$ *(resp.* $(v_0, \mathtt{in}^{<t}(u)) \in \mathsf{tr}^*$*), then* $(v_0, \mathtt{in}(u\sigma)) \in \mathsf{tr}$ *(resp.* $(v_0, \mathtt{in}^{<t}(u\sigma)) \in \mathsf{tr}$*).*
ii if $(a, \mathtt{out}(u)) \in \mathsf{tr}^*$ *for some* $a \in \mathcal{A}$*, then* $R\Phi_{\mathsf{sd}}\!\downarrow =_{\mathsf{E}} u\sigma$ *for some recipe* R*.*

Proof. Along an execution, variables occurring in input as well as those occurring in a let instruction are instantiated. We denote by τ_{tr} the substitution associated to a given execution tr. We now establish each item separately.

Item (i): We consider $\mathsf{V}(v_0, p_0) = \mathsf{a}_1.\ldots.\mathsf{a}_k$. Because $(v_0, \mathtt{in}^\star(u)) \in \mathsf{tr}^*$, we have that there exists $i_0 \in \{1, \ldots, k\}$ such that $a_{i_0} = \mathtt{in}^\star(x)$ for some $x \in \mathcal{X}$ and $x\tau_{\mathsf{tr}^*} = u$. By Lemma 3, we have that there exists $\tau : \mathcal{X} \mapsto \mathcal{N}$ a bijective renaming from variables to names freshly generated by P^* such that $u = x\theta_{\mathcal{P}}\!\downarrow\!\tau$.

On the other side, we know that the process has been entirely executed in tr and therefore there exists $(v_0, \mathtt{in}^\star(u')) \in \mathsf{tr}$ such that $u' = x\tau_{\mathsf{tr}}$ and thus by definition of $\theta_{\mathcal{P}}$, there exists σ such that $u' =_{\mathsf{E}} x\theta_{\mathcal{P}}\!\downarrow\!\sigma$. By consequence we have that $u' = x\tau_{\mathsf{tr}} =_{\mathsf{E}} x\theta_{\mathcal{P}}\!\downarrow\!\sigma = (u\tau^{-1})\sigma$. We conclude choosing $\sigma' = \tau^{-1}\sigma$.

Item (ii): We have that $(a, \mathtt{out}(u)) \in \mathsf{tr}^*$. We distinguish several cases depending on the origin of this output.

In case $a = v_0$, it is an immediate corollary of item (i). Indeed we can prove by induction on the length of tr that for each configuration K in exec, there exists a configuration K^* in exec^* such that if we note V^* the process executed by v_0 in K^* then $V^*\sigma$ is the process executed by v_0 in K.

Now, we assume that $a = p_0$, and we distinguish two cases depending on whether $u = m^j_{i_0+1}$ $(1 \leq j \leq k)$ or not. If not, applying Lemma 3, we have that $(v_0, \mathtt{in}(u)) \in \mathsf{tr}^*$ (or $(v_0, \mathtt{in}^{<t}(u)) \in \mathsf{tr}^*$), and applying item (i) we have know that $(v_0, \mathtt{in}(u\sigma)) \in \mathsf{tr}$ (or $(v_0, \mathtt{in}^{<t}(u\sigma)) \in \mathsf{tr}$). Therefore, we know that there exists a recipe R such that $R\Phi_{\mathsf{sd}}\!\downarrow =_E u\sigma$.

Now, we assume that $u = m^j_{i_0+1}$ for some $j \in \{1, \ldots, k\}$. Thanks to Lemma 3, we know that $(v_0, \mathtt{in}^{<2t_0}(m_{i_0+1})) \in \mathsf{tr}^*$ and $m_{i_0+1} = C_{\mathcal{P}}[m_{i_0}, m^1_{i_0+1}, \ldots, m^k_{i_0+1}] = x\theta_{\mathcal{P}}\!\downarrow\!\tau$ for some bijective renaming $\tau : \mathcal{X} \mapsto \mathcal{N}$ from variables to names freshly generated by P^*.

Applying item (i), we have that $(v_0, \mathtt{in}^{<2t_0}(m_{i_0+1}\sigma)) \in \mathsf{tr}$. Moreover, following the hypotheses on the structure of $\mathsf{V}(v_0, p_0)$, we know that in tr there is a unique output executed by v_0 before this guarded input that contains the challenge. In addition, p_0 cannot receive the challenge soon enough to make an output containing c available to fill the guarded input. Indeed, assume that it was the case, and let $t_{\mathtt{reset}}$ (resp. t_{out} and $t_{\mathtt{in}}$) the time when the $\mathtt{reset}$ action (resp. output of the challenge, reception of the guarded input) is executed in tr then we have that:

$$t_{\mathtt{in}} \geq t_{\mathsf{out}} + 2 \times \mathsf{Dist}_{\mathcal{T}_{\mathsf{oracle}}}(v_0, p_0) \geq t_{\mathtt{reset}} + 2 \times \mathsf{Dist}_{\mathcal{T}_{\mathsf{oracle}}}(v_0, p_0) = t_{\mathtt{reset}} + 2 \times t_0.$$

This is in contradiction with the constraint imposed by the guarded input: $t_{\mathtt{reset}} - t_{\mathtt{in}} < 2 \times t_0$. Finally, we deduce that there exists $\Phi^+ = \Phi \cup \{\mathsf{w} \xrightarrow{v_0, t_{\mathsf{out}}} c\}$ such that $c \notin st(img(\Phi))$ and a recipe R such that $R\Phi^+\!\downarrow =_{\mathsf{E}} m_{i_0+1}\sigma$. Lemma 5 applies and we conclude that there exists a recipe R_u such that $R_u\Phi^+\!\downarrow =_{\mathsf{E}} m^j_{i_0+1}\sigma$ and thus $R_u\Phi_{\mathsf{sd}}\!\downarrow =_{\mathsf{E}} u\sigma$ since Φ^+ is a subframe of Φ_{sd}. □

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 $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathit{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{MF}}$ 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:*

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

Proof. The direct implication is trivial, and we therefore concentrate on the other one. Let P_{sd} be a semi-dishonest prover for $\mathcal{P}_{\mathsf{prox}}$ with its associated frame Φ_{sd}, and $\mathtt{tr}$ be the trace witnessing this fact. We denote P^* the most general semi-dishonest prover, Φ^* its associated frame, and $\mathtt{tr}^*$ the trace witnessing this fact.

Applying Proposition 3, there exists a substitution $\sigma : \mathcal{N} \to \mathcal{T}(\Sigma_c^+, \mathcal{N} \cup \mathcal{A})$, from names freshly generated by P^* in $\mathtt{tr}^*$ to constructor terms such that:

i if $(v_0, \mathtt{in}(u)) \in \mathtt{tr}^*$, then $(v_0, \mathtt{in}(u\sigma)) \in \mathtt{tr}$.
ii if $(a, \mathtt{out}(u)) \in \mathtt{tr}^*$ for some $a \in \mathcal{A}$, then $R\Phi_{\mathsf{sd}}{\downarrow} =_{\mathsf{E}} u\sigma$ for some recipe R.

By hypothesis, there exist a topology $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$ and a valid initial configuration K^* for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^* \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$ such that

$$(\mathcal{P}_{\mathsf{init}}; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi^*; 0) \xrightarrow{\mathsf{tr}_0}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi^* \cup \Phi_{\mathsf{out}}; t)$$

for some frame Φ_{out}. Without loss of generality, we may assume that $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ with $\mathcal{M}_0 \neq \emptyset$. Otherwise, we add such a malicious agent, and the trace remains executable. Applying the substitution σ along this execution, we obtain a valid execution (remember that our calculus does not feature else branches):

$$(\mathcal{P}_{\mathsf{init}}\sigma; \Phi^{\mathcal{T}}_{\mathcal{I}_0}\sigma \cup \Phi^*\sigma; 0) \xrightarrow{\mathsf{tr}_0\sigma}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}\sigma; \Phi^{\mathcal{T}}_{\mathcal{I}_0}\sigma \cup \Phi^*\sigma \cup \Phi_{\mathsf{out}}\sigma; t).$$

Actually, since names occurring in $dom(\sigma)$ are names freshly generated by P^*, we have that $\mathcal{P}_{\mathsf{init}}\sigma = \mathcal{P}_{\mathsf{init}}$, $\Phi^{\mathcal{T}}_{\mathcal{I}_0}\sigma = \Phi^{\mathcal{T}}_{\mathcal{I}_0}$, and therefore, we have that:

$$(\mathcal{P}_{\mathsf{init}}; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi^*\sigma; 0) \xrightarrow{\mathsf{tr}_0\sigma}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}\sigma; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi^*\sigma \cup \Phi_{\mathsf{out}}\sigma; t).$$

Finally, from item 2, we have that for any $u \in img(\Phi^*)$, there exists a recipe R such that $R\Phi_{\mathsf{sd}}{\downarrow} =_{\mathsf{E}} u\sigma$. We can thus deduce that for any term v and recipe R_v such that $R_v(\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi^*\sigma){\downarrow} =_{\mathsf{E}} v$ we have that there exists R'_v such that $R'_v(\Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}){\downarrow} =_{\mathsf{E}} v$. Starting by applying a TIM rule with a delay δ equal to twice the greatest distance between two agents in $\mathcal{T}$, we have:

$$(\mathcal{P}_{\mathsf{init}}; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}}; 0) \xrightarrow{\mathsf{tr}_0\sigma}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t'}_{v_0} \uplus \mathcal{P}\sigma; \Phi^{\mathcal{T}}_{\mathcal{I}_0} \cup \Phi_{\mathsf{sd}} \cup \Phi'_{\mathsf{out}}\sigma; t + \delta).$$

with $\phi'_{\mathsf{out}} = \{\mathsf{w} \xrightarrow{a, t+\delta} u\sigma \mid \mathsf{w} \xrightarrow{a,t} u \in \Phi_{\mathsf{out}}\}$. The delay δ enables a dishonest agent (there is one by assumption) to build any term occurring in $\Phi^*\sigma$ from Φ_{sd}. □

Corollary 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be a DB protocol and $\mathcal{I}_0$ be a template such that $\mathcal{P}_{\mathsf{prox}}$ is $\mathcal{I}_0$-executable. Let P^* be the most general semi-dishonest prover for $\mathcal{P}_{\mathsf{prox}}$ 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:*

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

Proof. We distinguish the two directions of the implication.

If $\mathcal{P}_{\mathsf{prox}}$ is terrorist fraud resistant w.r.t. t_0-proximity then there exist a topology $\mathcal{T} \in \mathcal{C}_{\mathsf{MF}}$ and a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}$ and $\Phi^* \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$ such that:

$$K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Since $names(\Phi^*) \cap \mathcal{A} \subseteq \{v_0, p_0, e_1, e_2\}$, Lemma 2 applies and we obtain that there exits a valid initial configuration K_0 for $\mathcal{P}_{\mathsf{prox}}$ w.r.t. $\mathcal{T}^{t_0}_{\mathsf{MF}}$ and $\Phi^* \cup \Phi^{\mathcal{T}}_{\mathcal{I}_0}$ such that:

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

The other direction of the implication is an immediate application of Theorem 2 because $\mathcal{T}^{t_0}_{\mathsf{MF}} \in \mathcal{C}_{\mathsf{MF}}$.