Proving physical proximity using symbolic models

Alexandre Debant, Stéphanie Delaune, Cyrille Wiedling

Proving physical proximity using symbolic models *

Alexandre Debant, Stéphanie Delaune, and Cyrille Wiedling

Univ Rennes, CNRS, IRISA, France

Abstract. For many modern applications like *e.g.* contactless payment, and keyless systems, ensuring physical proximity is a security goal of paramount importance. Formal methods have proved their usefulness when analysing standard security protocols. However, existing results and tools do not apply to *e.g.* distance bounding that aims to ensure physical proximity between two entities. This is due in particular to the fact that existing models do not represent in a faithful way the locations of the participants, and the fact that transmission of messages takes time. In this paper, we propose several reduction results: when looking for an attack, it is actually sufficient to consider a simple scenario involving at most four participants located at some specific locations. An interesting consequence of our reduction results is that it allows one to reuse ProVerif, an automated tool developed for analysing standard security protocols. As an application, we analyse several distance bounding protocols, as well as a contactless payment protocol.

1 Introduction

The shrinking size of microprocessors as well as the ubiquity of wireless communication have led to the proliferation of portable computing devices with novel security requirements. Whereas traditional security protocols achieve their security goals relying solely on cryptographic primitives like encryptions and hash functions, this is not the case anymore for many modern applications like *e.g.* contactless payment. Actually, a typical attack against these devices is the so-called relay attack, as demonstrated for EMV in [16]. Such an attack allows a malicious participant to relay communication between a victim's card (possibly inside a wallet) and a genuine terminal so that the victim's card, even if it is far away from the terminal, will pay the transaction. Due to the contactless nature of most of our communication, obtaining reliable information regarding physical proximity is of paramount importance and specific protocols, namely distance bounding protocols, were proposed to achieve this specific goal [12, 24]. They typically take into account the round trip time of messages and the transmission velocity to infer an upper bound of the distance between two participants.

In the context of standard security protocols, such as key establishment protocols, formal methods have proved their usefulness for providing security guarantees or detecting attacks. The purpose of formal verification is to provide

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

rigorous frameworks and techniques to analyse protocols and find their flaws. For example, a flaw has been discovered in the Single-Sign-On protocol used *e.g.* by Google Apps. It has been shown that a malicious application could very easily get access to any other application (*e.g.* Gmail or Google Calendar) of their users [2]. This flaw has been found when analysing the protocol using formal methods, abstracting messages by a term algebra and using the Avantssar validation platform [3]. Another example is a flaw on vote-privacy discovered during the formal and manual analysis of an electronic voting protocol [19]. All these results have been obtained using formal symbolic models, where most of the cryptographic details are ignored using abstract structures, and the communication network is assumed to be entirely controlled by an omniscient attacker. The techniques used in symbolic models have become mature and several tools for protocol verification are nowadays available [9, 3, 28].

However, protocols whose security rely on constraints from the physical world fall outside the scope of traditional symbolic models that are based on the omniscient attacker who controls the entire network, and who can for instance relay messages without introducing any delay. Due to the lack of formal symbolic models and verification tools, distance bounding protocols are only analysed so far with respect to some specific attack types known as *e.g.* distance fraud, mafia fraud, and terrorist fraud. Recently, another type of attack, namely distance hijacking [20], has been discovered, and many protocols have been shown to be vulnerable to this new type of attacks. Following [8, 23], and more recently [25], our aim is to bridge the gap between informal approaches currently used to analyse these protocols and the formal approaches already used for analysing traditional security protocols.

Our contributions. We first propose a calculus which allows timed protocols as well as the notion of physical proximity to be formally described. We model cryptography as a black box, thus the attacker cannot break cryptography, *e.g.* decrypt a message without having the appropriate decryption key. To model timed protocols in an accurate way, our communication model is subject to physical restrictions. These constraints apply to honest agents and attackers. An attacker can only intercept messages at his location, and attackers can not instantaneously exchange their knowledge: transmitting messages takes time. This models reality, where the attackers' ability to observe and communicate messages depends on their locations.

Our main contribution is to provide reduction results in the spirit of the one obtained in [17] for traditional protocols: if there is an attack, then there is one considering only few participants at some specific locations. The results slightly differ depending on the type of attacks we consider (distance fraud, mafia fraud, or hijacking attack) but it allows one to focus on topologies involving a small number of participants (at most 4 including the malicious ones). We therefore reduce the number of topologies to be considered from infinitely many to only one (actually one per type of attacks). Our results hold in a rather general setting. In particular, we consider arbitrary cryptographic primitives as soon as they can be expressed using rewriting rules modulo an equational theory.

An interesting consequence of our reduction results is that it allows one to reuse techniques and tools developed for standard security protocols. Actually, we show how to encode these simple topologies, as well as the timing constraints, relying on the phase mechanism available in ProVerif. As an application, we analyse several distance bounding protocols, and we also consider the contactless payment protocol described in [16]. All files related to the case studies are available here:

http://people.irisa.fr/Alexandre.Debant/proving-physical-proximity-using-symbolic-methods.html

Related work. Until recently, most distance bounding protocols have been analysed without a formal approach. Recent efforts have been made on proving security of distance bounding protocols. For instance, in 2001, Avoine *et al.* [5] proposed a framework in which many protocols have been analysed and compared in a unified manner [4]. A rather general model has been proposed by Boureanu *et al.* in [11]. This computational model captures all the classical types of attacks and generalises them enabling attackers to interact with many provers and verifiers. These models are very different from ours. Indeed, we consider here a *formal symbolic model* in which messages are no longer bitstrings but they are abstracted away by terms. Some recent attempts have been made to design formal symbolic model suitable to analyse distance bounding protocols: e.g. a model based on multiset rewriting rules has been proposed in [8] and [25], another one based on strand spaces is available here [30]. Even if our model shares some similarities with those mentioned above, we choose to design a new one based on the applied pi calculus [1]. This allows us in particular to connect our theoretical results with the ProVerif verification tool that we ultimately use to analyse protocols.

Our main reduction result follows the spirit of [18] where it is shown that it is sufficient to consider five specific topologies when analysing routing protocols. To our knowledge, the only work proposing a reduction result suitable for distance bounding protocols is [30]. In this paper, the authors show that n attackers are actually sufficient when analysing an initial configuration involving at most n honest participants. Moreover, due to the way attackers are located (close to each honest participant), their result can not be applied to analyse some well-known attack scenarios (*e.g.* hijacking attack, distance fraud) that typically disallow the presence of an attacker in the neighbourhood of some honest participants. In contrast, our result reduces to only one topology, even when considering an arbitrary number of honest participants, and it applies to the scenarios mentioned above. A consequence of this result is that we can leverage the ProVerif tool to analyse such protocols. To do that we get some inspiration from [16] and we use the phase mechanism of ProVerif to encode timing constraints. Our contributions improve upon their work by providing a strong theoretical foundation to their idea. Moreover, in order to consider scenario in which attackers are far away, and thus unable to produce an answer within the delay, we slightly modify the tool to discard some attacker behaviours. This was needed to be able to obtain meaningful results using ProVerif when analysing distance fraud

and distance hijacking scenarios. Recently, a methodology to analyse distance bounding protocols within the Tamarin verification tool has been proposed [25]. Their model is not flexible enough to allow one to consider the different class of attacks, e.g. they cannot prove that a protocol is distance fraud resistant but vulnerable to mafia fraud. Nevertheless, they are able to analyse many protocols, confirmed some known vulnerabilities in a number of protocols, and discovered unreported attacks. All these protocols have been analysed in our framework, and a comparison is provided in Section 7.

2 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. This provides a lot of flexibility and allows one to model various cryptographic primitives.

2.1 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 for instance to represent nonces drawn by the attacker. We also consider 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 following signature $\Sigma_{\mathsf{ex}} = \Sigma_c \uplus \Sigma_d$:

$\Sigma_c = \{\mathsf{commit}, \mathsf{sign}, \mathsf{sk}, \mathsf{vk}, \mathsf{ok}, \langle\rangle, \oplus, 0\}$, $\Sigma_d = \{\mathsf{open}, \mathsf{getmsg}, \mathsf{check}, \mathsf{proj}_1, \mathsf{proj}_2, \mathsf{eq}\}$.

The symbols open and commit (arity 2) represent a commitment scheme, whereas the symbols sign, check (arity 2), getmsg, sk, and vk (arity 1) are used to model a signature scheme. Pairing is modelled using $\langle\ \rangle$ (arity 2), whereas projection functions are denoted proj_1 and proj_2 (arity 1). We also consider the symbols $\oplus$ and 0 to model the exclusive-or operator. Finally, we consider the function symbol eq to model equality test.

2.2 Equational theory

Following the approach developed in [10], constructor terms are subject to an *equational theory*. This allows one 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. Formally, $=_{\mathsf{E}}$ is the smallest congruence on constructor terms, which contains $u = v$ in E, and that is closed under substitutions of terms for variables.

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

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

2.3 Rewriting rules

As in [10], we also give a meaning to destructor symbols. This is done through a set of rewriting 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 rewriting rule $\mathsf{g}(t_1, \ldots, t_n) \to t$ such that $u|_p = \mathsf{g}(t_1, \ldots, t_n)\theta$ for some substitution θ. Moreover, we assume that $t_1\theta, \ldots, t_n\theta$ as well as $t\theta$ are messages. We only consider sets of rewriting rules that yield a *convergent* rewriting system, 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 Σ_{ex}, only sk is in Σ_{priv}. The properties of the symbols in Σ_d are reflected through the following rewriting rules:

$$\mathsf{check}(\mathsf{sign}(x, \mathsf{sk}(y)), \mathsf{vk}(y)) \to \mathsf{ok} \quad \mathsf{getmsg}(\mathsf{sign}(x, \mathsf{sk}(y))) \to x \quad \mathsf{proj}_1(\langle x_1, x_2 \rangle) \to x_1$$
$$\mathsf{eq}(x, x) \to \mathsf{ok} \quad \mathsf{open}(\mathsf{commit}(x, y), y) \to x \quad \mathsf{proj}_2(\langle x_1, x_2 \rangle) \to x_2.$$

3 Timed security protocols

In this section, we present our model which incorporates node location, time, and communication distance.

3.1 Process algebra

Protocols are modelled through processes using the following grammar:

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

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 $t \in \mathbb{R}_+$.

Most of these constructions are rather standard. As usual in symbolic models, 0 denotes the empty process that does nothing, and the `new` instruction is used to model fresh name generation. Then, we have standard constructions to model inputs and outputs. We may note the special construction $\mathtt{in}^{<t}(x)$ that combines an input with a constraint on the local clock of the agent executing this action. This construction is in contrast with the approach proposed in e.g. [25] where input actions are not subject to any timing constraint, and are therefore always possible provided that enough time has elapsed. From this point of view, our model represents the reality more faithfully since an agent will not proceed an input arriving later than expected. The `reset` instruction will reset the local clock of the agent. Finally, the process $\mathtt{let}\ x = v\ \mathtt{in}\ P$ tries to evaluate v, and in case of success the process P is executed; otherwise the process is blocked. In particular, this construction allows us to model the usual conditional operator relying on the destructor `eq`.

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). We consider *parametrised processes*, denoted $P(z_0, \ldots, z_n)$, where $z_0, \ldots, z_n$ are variables from a special set $\mathcal{Z}$ (disjoint from $\mathcal{X}$ and $\mathcal{W}$). Intuitively, these variables will be instantiated by agent names, and z_0 corresponds to the name of the agent that executes the process. A *role* $R = P(z_0, \ldots, z_n)$ is a parametrised process that does not contain any agent name, and such that $fv(R) \subseteq \{z_0, \ldots, z_n\}$. A *protocol* is then simply a set of roles.

Example 4. As a running example, we consider the signature-based Brands and Chaum distance bounding protocol [12] that is informally described below:

1. $P \to V : \mathsf{commit}(m, k)$
2. $V \to P : n$
3. $P \to V : n \oplus m$
4. $P \to V : k$
5. $P \to V : \mathsf{sign}(\langle n, n \oplus m \rangle, sk(P))$

The prover P generates a nonce m and a key k, and sends a commitment to the verifier V. The verifier V generates his own nonce n and initiates the time measurement phase, sometimed called the *rapid phase*. In this exchange, P answers with the exclusive-or of these two nonces. This step has to be done as quickly as possible since V will reject any answer arriving too late (a long response time does not give him any guarantee regarding its proximity with the prover). After this phase, P sends a means to open the commitment, as well as a signature on the values exchanged during the rapid phase. When a verifier ends a session of this protocol, the prover with whom he is communicating should be located in his neighbourhood. In our setting, the protocol is defined by the parametrised

processes given below:

$P(z_P) :=$
$\quad \texttt{new}\ m.\texttt{new}\ k.$
$\quad \texttt{out}(\mathsf{commit}(m,k)).$
$\quad \texttt{in}(x_n).$
$\quad \texttt{out}(x_n \oplus m).$
$\quad \texttt{out}(k).$
$\quad \texttt{out}(\mathsf{sign}(\langle x_n, x_n \oplus m\rangle, \mathsf{sk}(z_P))).0$

$V(z'_V, z'_P) :=$
$\quad \texttt{in}(y_c).\texttt{new}\ n.$
$\quad \texttt{reset}.\texttt{out}(n).\texttt{in}^{<2\times t_0}(y_0).$
$\quad \texttt{in}(y_k).\texttt{in}(y_{\mathsf{sign}}).$
$\quad \texttt{let}\ y_m = \mathsf{open}(y_c, y_k)\ \texttt{in}$
$\quad \texttt{let}\ y_{\mathsf{check}} = \mathsf{check}(y_{\mathsf{sign}}, \mathsf{vk}(z'_P))\ \texttt{in}$
$\quad \texttt{let}\ y_{\mathsf{eq}} = \mathsf{eq}(\langle n, n \oplus y_m\rangle, \mathsf{getmsg}(y_{\mathsf{sign}}))\ \texttt{in}\ 0.$

3.2 Configuration and topology

Each process has a location. As in the classical Dolev-Yao model [21], the attackers control the entire network but interacting with agents who are far away takes time. To formalise this, our execution model is parametrised by a topology.

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

- $\mathcal{A}_0 \subseteq \mathcal{A}$ *is the finite set of agents composing the system;*
- $\mathcal{M}_0 \subseteq \mathcal{A}_0$ *is the subset of agents that are dishonest;*
- $\mathsf{Loc}_0 : \mathcal{A}_0 \to \mathbb{R}^3$ *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 for which we analyse the security of the protocol.*

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 euclidian 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 $t - \mathsf{Dist}_{\mathcal{T}_0}(a,b)$.

Note that our model is not restricted to a single dishonest node. In particular, our results apply to the case of several compromised nodes that communicate (and therefore share their knowledge). However, communication is subject to physical constraints: message transmission takes time determined by the distance between nodes. All agents, including attackers, are subject to these constraints. This result in a distributed attacker with restricted, but more realistic, communication capabilities than those of the traditional Dolev-Yao attacker.

Our semantics is given by a transition system over configurations that manipulates *extended processes*, i.e. expressions of the form $\lfloor P_a \rfloor_a^{t_a}$ with $a \in \mathcal{A}$, P_a a process such that $fv(P_a) = \emptyset$, and $t_a \in \mathbb{R}_+$. Intuitively, P_a describes the actions of agent a, and t_a his local clock. In order to store the messages that have been outputted so far, we extend the notion of *frame* (introduced in [1]) to keep track of the time at which the message has been outputted and by whom.

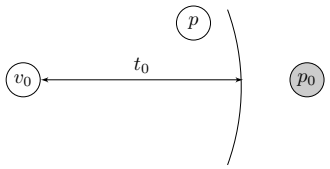
Definition 2. *Given a topology* $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$*, a* configuration K *over* $\mathcal{T}_0$ *is a tuple* $(\mathcal{P}; \Phi; t)$*, where:*

- $\mathcal{P}$ *is a multiset of* extended process $\lfloor P \rfloor_a^t$ *with* $a \in \mathcal{A}_0$*;*
- $\Phi = \{\mathsf{w}_1 \xrightarrow{a_1,t_1} u_1, \ldots, \mathsf{w}_n \xrightarrow{a_n,t_n} u_n\}$ *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.*

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 5. Continuing Example 4, a topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ is depicted on the right where $\mathcal{A}_0 = \{p_0, v_0, p\}$, and $\mathcal{M}_0 = \{p_0\}$. The precise location of each agent is not relevant, only the distance between them matters. Here, we assume that $\mathsf{Dist}_{\mathcal{T}_0}(p, v_0) < t_0$ whereas $\mathsf{Dist}_{\mathcal{T}_0}(p_0, v_0) \geq t_0$. A typical configuration is:



$$K_0 = (\lfloor P(p) \rfloor_p^0 \uplus \lfloor V(v_0, p_0) \rfloor_{v_0}^0 ; \{\mathsf{w}_1 \xrightarrow{p_0,0} \mathsf{sk}(p_0)\}; 0)$$

where p is playing the role of the prover and v_0 the role of the verifier with a dishonest agent p_0. The extended frame only contains the signature key of the dishonest agent, i.e. $\mathsf{sk}(p_0)$. A more realistic configuration would include other instances of these two roles and will give more knowledge to the attacker, but we will see that this configuration is already sufficient to present an attack.

3.3 Semantics

Given a topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$, the operational semantics of processes is formally defined by the rules given in Figure 1. The TIM rule allows time to elapse meaning that the global clock as well as the local clocks will be shifted by δ as formally defined below:

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

The RST rule allows an agent to reset his local clock. The other rules are rather standard. The IN rule allows an agent a to evolve when receiving a message. Note that, such a message has necessarily been sent at time t_b by some other agent b who has to be in possession of all the necessary information at that time. We sometimes simply write $\rightarrow_{\mathcal{T}_0}$ instead of $\xrightarrow{a,\alpha}_{\mathcal{T}_0}$. The relation $\rightarrow_{\mathcal{T}_0}^*$ is the reflexive and transitive closure of $\rightarrow_{\mathcal{T}_0}$, and we often write $\xrightarrow{\mathsf{tr}}_{\mathcal{T}_0}$ to emphasise the sequence of labels tr that has been used during this execution.

Example 6. Continuing Example 5, the following sequence of transitions is enabled from the configuration K_0:

$$K_0 \xrightarrow{p,\tau}_{\mathcal{T}_0} \xrightarrow{p,\tau}_{\mathcal{T}_0} \xrightarrow{p,\mathsf{out}(\mathsf{commit}(m',k'))}_{\mathcal{T}_0} \rightarrow_{\mathcal{T}_0} \xrightarrow{v_0,\mathtt{in}(\mathsf{commit}(m',k'))}_{\mathcal{T}_0} \xrightarrow{v_0,\tau}_{\mathcal{T}_0} \xrightarrow{v_0,\tau}_{\mathcal{T}_0} K_1$$

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

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

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

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

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

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

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

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

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

Fig. 1: Semantics of our calculus

where $K_1 = (\lfloor P_1 \rfloor_p^{\delta_0} \uplus \lfloor V_1 \rfloor_{v_0}^{0}; \{\mathsf{w}_1 \xrightarrow{p_0,0} \mathsf{sk}(p_0),\ \mathsf{w}_2 \xrightarrow{p,0} \mathsf{commit}(m',k')\}; \delta_0)$. The processes located in p and v_0 have evolved. We have that:

- $P_1 = \mathtt{in}(x_n).\mathtt{out}(x_n \oplus m').\mathtt{out}(k').\mathtt{out}(\mathsf{sign}(\langle x_n, x_n \oplus m' \rangle, \mathsf{sk}(p)))$; and
- $V_1 = \mathtt{out}(n').\mathtt{in}^{<2\times t_0}(y_0).\mathtt{in}(y_k).\mathtt{in}(y_{\mathsf{sign}}).\mathtt{let}\ y_m = \mathsf{open}(y_c, y_k)\ \mathtt{in} \ldots$

This corresponds to the beginning of a normal execution between p and v_0. The two first transitions have been triggered using the NEW rule. The third one is an instance of the rule OUT. The message outputted at location p is received at location v_0, but an instance of the rule TIM (here with $\delta_0 = \mathsf{Dist}_{\mathcal{T}_0}(p, v_0)$) is needed to allow the message to reach this location.

4 Security properties

A distance bounding protocol is a protocol in which a party (the verifier) is assured of the identity of another party (the prover), as well as the fact that this prover is located in his neighbourhood. Several frauds against distance bounding protocols are usually considered. We introduce in Section 4.2 the ones that will be studied in this paper, and we explain how they will be formalised. Before to do that, we introduce the notion of t_0-proximity and the notion of valid initial configuration that aims to represent all the scenarios that have to be analysed once the topology has been fixed.

4.1 Physical proximity on a given topology

For the sake of simplicity, we assume that processes representing instances of distance bounding protocols contain a process (typically a session of the verifier) that ends with a special action of the form $\mathsf{end}(v, p)$. Checking whether a protocol ensures physical proximity w.r.t. a given configuration K_0 is defined as follows.

Definition 3. *Let $\mathcal{T}_0$ be a topology and K_0 be a configuration over $\mathcal{T}_0$. We say that K_0 admits* an attack w.r.t. t_0-proximity *in $\mathcal{T}_0$ if*

$$K_0 \to^*_{\mathcal{T}_0} (\lfloor \mathsf{end}(a_1, a_2) \rfloor_a^{t_a} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathit{Dist}_{\mathcal{T}_0}(a_1, a_2) \geq t_0.$$

Of course, when analysing a distance bounding protocol, not all the configurations are interesting. Depending on the type of fraud under consideration, we explain in Definition 4 the configurations that are initial valid configurations, and therefore have to be analysed before declaring whether a protocol is secure or not.

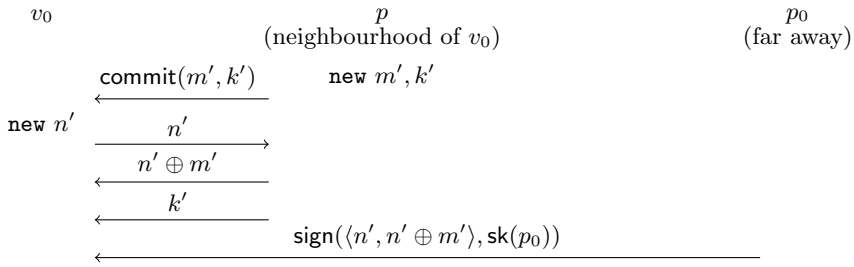


Fig. 2: Distance hijacking attack on the Brands and Chaum's protocol

Example 7. Continuing Example 5, we consider the configuration K_0' below:

$$K_0' = (\lfloor P(p) \rfloor_p^0 \uplus \lfloor V'(v_0, p_0) \rfloor_{v_0}^0 ; \{\mathsf{w}_1 \xrightarrow{p_0,0} \mathsf{sk}(p_0)\}; 0)$$

where $V'(z_V', z_P')$ is $V(z_V', z_P')$ in which the occurrence of the process null has been replaced by $\mathsf{end}(z_V', z_P')$. The sequence of transitions described in Example 6 is still possible ending in

$$K_1' = (\lfloor P_1 \rfloor_p^{\delta_0} \uplus \lfloor V_1' \rfloor_{v_0}^0 ; \{\mathsf{w}_1 \xrightarrow{p_0,0} \mathsf{sk}(p_0),\ \mathsf{w}_2 \xrightarrow{p,0} \mathsf{commit}(m', k')\}; \delta_0)$$

where V_1' is V_1 in which the occurrence of the process null has been replaced by $\mathsf{end}(v_0, p_0)$. Now, we can pursue this execution as follows:

$$K_1' \xrightarrow{v_0,\mathtt{out}(n')} \mathcal{T}_0 \to \mathcal{T}_0 \xrightarrow{p,\mathtt{in}(n')} \mathcal{T}_0 \xrightarrow{p,\mathtt{out}(n' \oplus m')} \mathcal{T}_0 \xrightarrow{p,\mathtt{out}(k')} \mathcal{T}_0 \to \mathcal{T}_0 \xrightarrow{v_0,\mathtt{in}^{<2\times t_0}(n' \oplus m')} \mathcal{T}_0 \xrightarrow{v_0,\mathtt{in}(k')} \to \mathcal{T}_0 \xrightarrow{v_0,\mathtt{in}(\mathsf{sign}(\langle n', n' \oplus m' \rangle, \mathsf{sk}(p_0)))} (\lfloor P_2 \rfloor_p^{3\delta_0 + 2\delta_0'} \uplus \lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{2\delta_0 + 2\delta_0'} ; \Phi; 3\delta_0 + 2\delta_0')$$

The two first lines correspond to a normal execution of the protocol between v_0 and p. Note that, on each line, we need an instance of the TIM rule with $\delta_0 = \mathsf{Dist}_{\mathcal{T}_0}(v_0, p) = \mathsf{Dist}_{\mathcal{T}_0}(p, v_0)$ to allow the sent message to reach its destination. The last transition does not follow the normal execution of the protocol. Actually, the dishonest agent p_0 is responsible of this input. He built this message from the messages n' and $n' \oplus m'$ that have been sent on the network, and the key $\mathsf{sk}(p_0)$ that is part of his initial knowledge. Note that he has to wait the necessary amount of time to allow these messages to reach him (e.g. $\delta'_0 = \mathsf{Dist}_{\mathcal{T}_0}(v_0, p_0)$), and some time is needed for the forged message to reach v_0 (actually $\delta'_0 = \mathsf{Dist}_{\mathcal{T}_0}(v_0, p_0)$). Therefore, the first rule of the last line is an instance of the TIM rule during which a delay of $2\delta'_0$ has elapsed.

Note that, according to Definition 3, this corresponds to an attack w.r.t. t_0-proximity on configuration K'_0. Actually, this is the hijacking attack that has been reported in [20] and described in Figure 2. Here, a dishonest prover p_0 exploits an honest prover p (located in the neighbourhood of v_0) to provide the verifier v_0 with false information about the distance between p_0 and v_0.

Once the topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ is fixed, when analysing whether a protocol is secure or not, we have to consider any configuration that is valid w.r.t. the given topology. We consider a protocol $\mathcal{P}_{\mathsf{prox}}$, and we assume that the initial knowledge of dishonest participants is given through a template $\mathcal{I}_0$, i.e. a set of terms in $\mathcal{T}(\Sigma_c^+, \mathcal{Z})$. Using this template $\mathcal{I}_0$, and considering a set of agents $\mathcal{A}_0$, we derive the initial knowledge of agent $a \in \mathcal{A}_0$ as follows:

$$\mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_0) = \{(u_0\{z_0 \mapsto a\})\sigma \text{ ground } \mid u_0 \in \mathcal{I}_0 \text{ and } img(\sigma) \subseteq \mathcal{A}_0\}.$$

Definition 4. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the special action $\mathsf{end}(z_0, z_1)$, $\mathcal{I}_0$ be a template, and $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology. A configuration $K = (\mathcal{P}; \Phi; t)$ is a* valid initial configuration *for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}_0$ and $\mathcal{I}_0$ if:*

1. *$\mathcal{P} = \lfloor V_0(v_0, p_0) \rfloor_{v_0}^{t'} \uplus \mathcal{P}'$ for some t' and for each $\lfloor P' \rfloor_{a'}^{t'} \in \mathcal{P}'$ there exists $P(z_0, \ldots, z_k) \in \mathcal{P}_{\mathsf{prox}}$, and $a_1, \ldots, a_k \in \mathcal{A}_0$ such that $P' = P(a', a_1, \ldots, a_k)$.*
2. *$img(\lfloor \Phi \rfloor_a^t) = \mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_0)$ when $a \in \mathcal{M}_0$, and $img(\lfloor \Phi \rfloor_a^t) = \emptyset$ otherwise.*

When $t = 0$ and $t' = 0$ for each $\lfloor P' \rfloor_{a'}^{t'} \in \mathcal{P}$ and each $\{\mathsf{w} \xrightarrow{a',t'} u\} \in \Phi$, we say that K is a zero-configuration.

The first condition says that we consider initial configurations made up of instances of the roles of the protocols, and we only consider roles executed by agents located at the right place. The second condition allows one to give some initial knowledge to each malicious node. We may note that we do not give so much constraints regarding time. It is indeed important that all the possible initial configurations are analysed before declaring a protocol secure.

Example 8. Going back to Example 7 and considering the template $\mathcal{I}_0 = \{\mathsf{sk}(z_0)\}$, we have that K'_0 is a valid initial zero-configuration.

Definition 5. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the special action $\mathsf{end}(z_0, z_1)$, $\mathcal{I}_0$ be a template, and $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology. We say that $\mathcal{P}_{\mathsf{prox}}$ admits an attack w.r.t. t_0-proximity in $\mathcal{T}_0$ if there exists a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}_0$ and $\mathcal{I}_0$ such that K admits an attack w.r.t. t_0-proximity in $\mathcal{T}_0$.*
We say that $\mathcal{P}_{\mathsf{prox}}$ admits a zero-attack when in addition K is a zero-configuration.

When analysing a protocol, it is generally sufficient to consider valid initial configurations that are zero-configurations.

Proposition 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the action $\mathsf{end}(z_0, z_1)$, $\mathcal{I}_0$ be a template, and $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology. Moreover, we assume that any guarded input in $\mathcal{P}_{\mathsf{prox}}$ and V_0 is preceded by a* `reset`*. We have that $\mathcal{P}_{\mathsf{prox}}$ admits an attack w.r.t. t_0-proximity in $\mathcal{T}_0$ if, and only if, $\mathcal{P}_{\mathsf{prox}}$ admits a zero-attack w.r.t. t_0-proximity in $\mathcal{T}_0$.*

4.2 Classification of attacks

We consider different types of attacks as it is usually done in distance bounding protocols. We do not consider here the notion of terrorist fraud since this notion relies on the fact that an attacker located in the neighbourhood of the verifier helps the prover located outside the neighbourhood. This may require the dishonest prover to share some knowledge with the attacker but without giving him any advantage for future attacks (in particular without revealing him all his credential). This does not fit in our model since here we assume colluding attackers who share their knowledge (provided that enough time has elapsed).

Distance fraud. A *distance fraud* is an attack where a dishonest prover who is far away succeeds in convincing an honest verifier that he is actually close to him. In general, it is supposed that this dishonest prover cheats without help of other entities located in the neighbourhood of the verifier. Therefore, when considering distance fraud, we only have to consider topologies $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ as follows:

$$p_0 \in \mathcal{M}_0,\ v_0 \in \mathcal{A}_0 \setminus \mathcal{M}_0, \text{ and } \mathsf{Dist}_{\mathcal{T}_0}(v_0, a) \geq t_0 \text{ for any } a \in \mathcal{A}_0 \setminus \{v_0, p_0\}.$$

We denote $\mathcal{C}_{\mathsf{DF}}$ the set of topologies that satisfy these requirements. We may note that the valid initial configurations corresponding to such a topology will allow the agent v_0 to execute any role of the protocol an arbitrary number of times. This corresponds to a more complex scenario than those usually analysed in the context of distance fraud. Therefore, we will also consider the notion of *simple distance fraud* for which the agent v_0 is only authorised to execute the parametrised role $V_0(z_0, z_1)$.

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 external attacker. This attacker performs a man-in-the-middle attack between the verifier and the prover with the aim of convincing the verifier that the honest prover is actually close to it. Therefore, when considering mafia fraud, we consider topologies $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ that are as follows:

$$\text{both } v_0 \text{ and } p_0 \text{ are honest, i.e. } v_0, p_0 \in \mathcal{A}_0 \setminus \mathcal{M}_0.$$

We denote $\mathcal{C}_{\mathsf{MF}}$ the set of topologies that satisfy these requirements.

Distance hijacking fraud. A *distance hijacking fraud* is an attack in which a dishonest prover located far away succeeds in convincing an honest verifier that he is actually close to him. Contrary to the notion of distance fraud, the dishonest prover may exploit honest entities located in the neighbourhood of the verifier. Therefore, we only have to consider topologies $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ that are as follows:

$$p_0 \in \mathcal{M}_0,\ v_0 \in \mathcal{A}_0 \setminus \mathcal{M}_0, \text{ and } \mathsf{Dist}_{\mathcal{T}_0}(v_0, a) \geq t_0 \text{ for any } a \in \mathcal{M}_0.$$

We denote $\mathcal{C}_{\mathsf{DH}}$ the set of topologies that satisfy these requirements.

Definition 6. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the special event $\mathsf{end}(z_0, z_1)$, and $\mathcal{I}_0$ be a template. We say that $\mathcal{P}_{\mathsf{prox}}$ admits a distance fraud attack (resp. mafia fraud attack, distance hijacking attack) w.r.t. t_0-proximity if there exist $\mathcal{T} \in \mathcal{C}_{\mathsf{DF}}$ (resp. $\mathcal{C}_{\mathsf{MF}}$, $\mathcal{C}_{\mathsf{DH}}$), a valid initial configuration K for $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$ such that K admits an attack w.r.t. t_0-proximity in $\mathcal{T}$.*

Moreover, in case $\mathcal{T} \in \mathcal{C}_{\mathsf{DF}}$ and V_0 is the only process located in v_0 in the configuration K, then such an attack is called a simple distance fraud attack.

Our main contribution is to provide reduction results that allow one to analyse the security of a protocol w.r.t. the frauds mentioned above considering only a specific topology. Then, we will show how to leverage an existing tool ProVerif to automatically analyse this notion of physical proximity.

5 Reducing the topology

Our reduction results allow one to analyse the security of a protocol considering only a specific topology. Each result is specific to a particular fraud but their proofs rely on some common ingredients. In each case, we succeed in providing a unique topology on which it is actually sufficient to perform the security analysis. These topologies are rather simple (at most four agents) and are depicted below for each type of fraud.

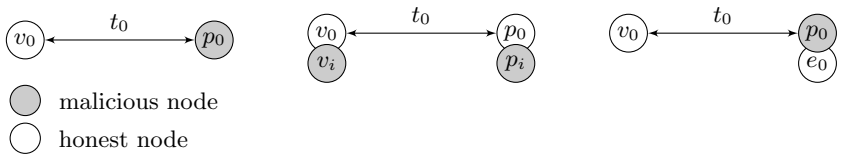


Fig. 3: Topologies $\mathcal{T}^{t_0}_{\mathsf{DF}}$, $\mathcal{T}^{t_0}_{\mathsf{MF}}$, and $\mathcal{T}^{t_0}_{\mathsf{DH}}$

5.1 Distance fraud

Regarding distance fraud w.r.t. t_0-proximity, the topology $\mathcal{T}^{t_0}_{\mathsf{DF}}$ is as follows:

$$\mathcal{T}^{t_0}_{\mathsf{DF}} = (\{p_0, v_0\}, \{p_0\}, \mathsf{Loc}_{\mathsf{DF}}, v_0, p_0) \text{ with } \mathsf{Dist}_{\mathcal{T}^{t_0}_{\mathsf{DF}}}(p_0, v_0) = t_0.$$

This topology is not uniquely defined. However, since our semantics only depends on the distance between agents (and not on their locations), it should be clear that we can consider an arbitrary one that fulfills the conditions expressed above.

Theorem 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the special event $\mathsf{end}(z_0, z_1)$, and $\mathcal{I}_0$ be a template. We have that $\mathcal{P}_{\mathsf{prox}}$ admits a distance fraud attack w.r.t. t_0-proximity, if and only if, there is an attack against t_0-proximity in the topology $\mathcal{T}^{t_0}_{\mathsf{DF}}$.*

Proof. (Sketch) We consider an attack trace in $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{DF}}$.

$$K_0 \to^*_{\mathcal{T}} (\, \lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0$$

We proceed in two main steps:

1. We show that the same trace can be done in a topology where all the agents in $\mathcal{A}_0$ are moved into an existing location in $\mathcal{T}^{t_0}_{\mathsf{DF}}$. More precisely, all the agents except v_0 are moved at $\mathsf{Loc}_{\mathsf{DF}}(p_0)$ whereas v_0 is moved at $\mathsf{Loc}_{\mathsf{DF}}(v_0)$. We may note that the distance between any two agents has been shorten (keeping p_0 and v_0 far away), and thus the trace can still be done.
2. Then, we apply a renaming of agents preserving their location. The trace is still executable, and we show that the resulting initial configuration (the one after application of the renaming) is still a valid initial configuration. □

In particular, the renaming used in the second step of the proof will leave v_0 unchanged. Therefore, a valid initial configuration built on $\mathcal{T}$ w.r.t. the simple distance fraud scenario will be transformed into a valid initial configuration built on $\mathcal{T}^{t_0}_{\mathsf{DF}}$ w.r.t. the simple distance fraud scenario, i.e. $V_0(v_0, p_0)$ will be the only process located in v_0.

5.2 Mafia fraud

Regarding mafia fraud, $\mathcal{T}^{t_0}_{\mathsf{MF}}$ is as follows: $\mathcal{T}^{t_0}_{\mathsf{MF}} = (\mathcal{A}_{\mathsf{MF}}, \mathcal{M}_{\mathsf{MF}}, \mathsf{Loc}_{\mathsf{MF}}, v_0, p_0)$ with $\mathcal{A}_{\mathsf{MF}} = \{p_0, v_0, p_i, v_i\}$, $\mathcal{M}_{\mathsf{MF}} = \{p_i, v_i\}$, $\mathsf{Loc}_{\mathsf{MF}}(p_0) = \mathsf{Loc}_{\mathsf{MF}}(p_i)$, $\mathsf{Loc}_{\mathsf{MF}}(v_0) = \mathsf{Loc}_{\mathsf{MF}}(v_i)$, and $\mathsf{Dist}_{\mathcal{T}^{t_0}_{\mathsf{MF}}}(p_0, v_0) = t_0$.

However, we may note that the reduction we did in case of distance fraud is not possible anymore since it will require to lengthen the distance between some nodes. Typically a node n in the neighborhood of v_0 would be shifted at the same location as v_0 but this would lengthen the distance between n and p_0. Since dishonest participants are allowed in the neighbourhood of v_0, getting some inspiration from [30], we consider a dishonest participant right next to each honest participant. Such a dishonest participant is ideally located to forge and send messages that will be received by the honest agent close to him.

However, contrary to the result provided in [30], our goal is not only to reduce the number of dishonest agents but also the number of honest agents that are involved in an attack trace. In order to ensure that moving (and reducing) the honest agents will not cause any trouble, we need an extra assumption. We require that each role of the protocol is executable. This is a reasonable assumption that will allow one to discard any role executed by a malicious participant. Intuitively, all these operations will be done directly by the attacker.

Definition 7. *Given a template $\mathcal{I}_0 = \{u_1, \ldots, u_k\}$, we say that a parametrised role $P(z_0, ..., z_n)$ is* $\mathcal{I}_0$-executable *if $fv(P) \subseteq \{z_0, ..., z_n\}$, $fn(P) = \emptyset$ and for any term u (resp. v) occurring in an output or a let construction, there exists $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, \{\mathsf{w}_1, \ldots, \mathsf{w}_k\} \uplus \mathcal{N} \uplus \mathcal{X})$ such that $u = R\sigma\downarrow$ (resp. $v\downarrow = R\sigma\downarrow$) where $\sigma = \{\mathsf{w}_1 \mapsto u_1, \ldots, \mathsf{w}_k \mapsto u_k\}$.*

A protocol $\mathcal{P}$ is $\mathcal{I}_0$-*executable* if each role of $\mathcal{P}$ is $\mathcal{I}_0$-executable.

Example 9. Going back to our running example given in Example 4. We have that $P(z_0)$ is $\{\mathsf{sk}(z_0)\}$-executable whereas $V(z_0, z_1)$ is $\{z_1\}$-executable. Therefore, we have that the protocol made of these two roles is $\{\mathsf{sk}(z_0), z_1\}$-executable.

Theorem 2. *Let $\mathcal{I}_0$ be a template, $\mathcal{P}_{\mathsf{prox}}$ be a protocol $\mathcal{I}_0$-executable, and $V_0(z_0, z_1)$ be a parametrised role containing the special event $\mathsf{end}(z_0, z_1)$. We have that $\mathcal{P}_{\mathsf{prox}}$ admits a mafia fraud attack w.r.t. t_0-proximity, if and only if, there is an attack against t_0-proximity in the topology $\mathcal{T}^{t_0}_{\mathsf{MF}}$.*

Proof. (Sketch) We consider an attack trace in $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{MF}}$.

$$K_0 \to^*_{\mathcal{T}} (\, \lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_v}_{v_0} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

We proceed in three main steps:

1. We reduce the number of active agents (those that are actually executing a process) - we do this for honest and malicious agents. We transform honest agent (but v_0 and p_0) into malicious ones. This intuitively gives more power to the attacker, and malicious agents in the neighborhood of v_0 are allowed in a mafia fraud scenario. Then, relying on our executability condition, we discard processes executed by malicious agents. These actions can actually be mimicked by an attacker located at the same place.
2. We reduce the number of attackers by placing them ideally (one close to each honest agent). Since we have removed all honest agents but two, we obtain a topology with only two dishonest agents.

3. To conclude, as in the case of a distance fraud attack, we reduce the knowledge showing that we can project all the dishonest agents that are located in p_0 on p_i and all the dishonest agents that are located in v_0 on v_i. □

5.3 Distance hijacking attack

First, we may note that the reduction we did in case of distance fraud or mafia fraud are not possible anymore. Clearly, we need to consider honest participants in the neighbourhood of the verifier and moving them on the same location as v_0 will lengthen the distance with p_0. Moreover, there is no hope to reduce the number of attackers by placing them close to each honest participant since this will introduce a malicious node in the neighbourhood of v_0. This is forbidden when considering a distance hijacking scenario. Actually, adding such a dishonest node in the neighbourhood of v_0 will always introduce a false attack since in our model dishonest participants share their knowledge. Therefore, this dishonest participant would be able to play the role of the prover as if he was the dishonest prover p_0 (who is actually far away).

Nevertheless, we will show that under reasonable conditions, we can reduce towards the topology $\mathcal{T}^{t_0}_{\mathsf{DH}}$ which is made up of 3 agents. More formally, $\mathcal{T}^{t_0}_{\mathsf{DH}} = (\mathcal{A}_{\mathsf{DH}}, \mathcal{M}_{\mathsf{DH}}, \mathsf{Loc}_{\mathsf{DH}}, v_0, p_0)$ with $\mathcal{A}_{\mathsf{DH}} = \{p_0, v_0, e_0\}$, $\mathcal{M}_{\mathsf{DH}} = \{p_0\}$, $\mathsf{Loc}_{\mathsf{DH}}(p_0) = \mathsf{Loc}_{\mathsf{DH}}(e_0)$, and $\mathsf{Dist}_{\mathcal{T}^{t_0}_{\mathsf{DH}}}(p_0, v_0) = t_0$.

Given a process P, we denote $\overline{P}$ the process obtained from P by removing `reset` instructions, and replacing all the occurrences of $\mathtt{in}^{<t}(x)$ by $\mathtt{in}(x)$.

Theorem 3. *Let $\mathcal{I}_0$ be a template, $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $t_0 \in \mathbb{R}_+$, and $V_0(z_0, z_1)$ be a parametrised role obtained using the following grammar:*

$$\begin{array}{rl} P, Q := \mathsf{end}(z_0, z_1) & \mid \mathtt{in}(x).P \quad \mid \mathtt{let}\ x = v\ \mathtt{in}\ P \\ \mid\ \mathtt{new}\ n.P & \mid \mathtt{out}(u).P \mid \mathtt{reset}.\mathtt{out}(u').\mathtt{in}^{<t}(x).P \end{array}$$

where $x \in \mathcal{X}$, $n \in \mathcal{N}$, $u, u' \in \mathcal{T}(\Sigma_c^+, \mathcal{X} \cup \mathcal{N} \cup \{z_0, z_1\})$, $v \in \mathcal{T}(\Sigma^+, \mathcal{X} \cup \mathcal{N} \cup \{z_0, z_1\})$ and $t \leq 2 \times t_0$. If $\mathcal{P}_{\mathsf{prox}}$ admits a distance hijacking attack w.r.t. t_0-proximity, then $\overline{\mathcal{P}_{\mathsf{prox}}}$ admits an attack against t_0-proximity in the topology $\mathcal{T}^{t_0}_{\mathsf{DH}}$.

In order to establish such a result, the idea is to move honest participants in the neighbourhood of v_0 at the same location as v_0. However, this will lengthen the distance between such an agent and those that are far away from v_0, and thus some messages will be forgeable only after some extra delay. This can be a problem in case such a message is useful to pass the guarded input. Thus, we will first transform the initial attack trace into an "attack" trace in an untimed model. This model (with no timing constraints to fullfill) is more suitable to reorder some actions in the trace. We will show in a second step how to come back in the original timed model.

As briefly explained above, we consider the untimed configuration associated to a configuration $K = (\mathcal{P}; \Phi; t)$. Formally, we have $\mathsf{untimed}(K) = (\mathcal{P}'; \Phi')$ with:

– $\mathcal{P}' = \{ \lfloor P \rfloor_a \mid \lfloor P \rfloor_a^t \in \mathcal{P} \text{ for some } t\}$, and

– $\Phi' = \{w \xrightarrow{a} u \mid \mathsf{w} \xrightarrow{a,t} u \in \Phi \text{ for some } t\}$.

Then, we consider a *relaxed semantics* over untimed configurations: $K \overset{a,\alpha}{\rightsquigarrow}_{\mathcal{T}} K'$ if there exist K_0 and K_0' such that $K_0 \xrightarrow{a,\alpha}_{\mathcal{T}} K_0'$ (for some rule other than the TIM rule), and for which $K = \mathsf{untimed}(K_0)$ (resp. $K' = \mathsf{untimed}(K_0')$).

Under the same hypotheses as those stated in Theorem 3, we establish a result that allows one to "clean" an attack trace by pushing instructions (before or after) outside the rapid phase delimited by a `reset` and its following guarded input. In the resulting trace, the only remaining actions in the rapid phase are those performed by agents who are close to v_0.

Proposition 2. *Let K_0 be a valid initial configuration for $\overline{\mathcal{P}_{\mathsf{prox}}}$ and V_0 w.r.t. a topology $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ and $\mathcal{I}_0$. If $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K_1$ then there exists an execution $K_0' \overset{\mathsf{tr}'}{\rightsquigarrow} K_1'$ such that $K_i' = \mathsf{untimed}(K_i)$ for $i \in \{0,1\}$.*

Moreover, for any sub-execution of $K_0' \overset{\mathsf{tr}'}{\rightsquigarrow} K_1'$ of the form

$$(\lfloor \mathtt{reset}.P \rfloor_{v_0} \uplus \mathcal{P}; \Phi_{\mathtt{reset}}) \overset{v_0,\tau}{\rightsquigarrow} (\lfloor P \rfloor_{v_0} \uplus \mathcal{P}; \Phi_{\mathtt{reset}}) \overset{\mathsf{tr}_0'}{\rightsquigarrow} K_{\mathtt{in}}^{-} \overset{v_0,\mathtt{in}^{<t}(u)}{\rightsquigarrow} K_{\mathtt{in}}'$$

where tr_0' only contains actions (a,α) with $\alpha \in \{\tau, \mathtt{out}(u), \mathtt{in}(u)\}$, we have that:

– *$2\mathit{Dist}_{\mathcal{T}}(v_0, a) < t$ for any $(a,\alpha) \in \mathsf{tr}_0'$;*
– *for any $(a, \mathtt{in}^{\star}(v))$ occurring in $\mathsf{tr}_0'.(v_0, \mathtt{in}^{<t}(u))$, the agent b responsible of the output and the recipe R (as defined in Figure 1) are such that either $2\mathit{Dist}_{\mathcal{T}}(v_0, b) < t$, or $vars(R) \subseteq dom(\Phi_{\mathtt{reset}})$.*

We are now able to prove our result regarding distance hijacking frauds.

Proof. (Sketch)

1. We start by removing `reset` instructions and by transforming any guarded input (but those in V_0) into simple inputs. The resulting trace is still an attack trace w.r.t. $\overline{\mathcal{P}_{\mathsf{prox}}}$.
2. Then, we apply Proposition 2 in order to obtain an attack trace in the relaxed semantics. We will exploit the extra conditions given by Proposition 2 in order to lift the trace in the timed model at step 4.
3. We now consider another topology $\mathcal{T}'$ with two locations (as $\mathcal{T}_{\mathsf{DH}}^{t_0}$) and such that agents close to v_0 are now located with v_0, and those that are far away from v_0 in $\mathcal{T}$ are now located with p_0. This execution is still a valid trace in $\mathcal{T}'$ since we consider the relaxed semantics.
4. Then, to lift this execution trace into our timed model, the basic idea is to wait enough time before a `reset` instruction to allow messages to be received by all the participants before starting the rapid phase.
5. To conclude, as in the previous attack scenarios, we reduce the initial knowledge and the number of agents by applying a renaming on agent names. □

$$\textsc{In}\quad (i : \mathtt{in}(x).P \uplus \mathcal{P}; \phi; i) \xrightarrow{\mathtt{in}(u)} (i : P\{x \mapsto u\} \uplus \mathcal{P}; \phi; i) \quad \text{where } R \text{ is a recipe such that } R\phi\downarrow = u \text{ is a message.}$$

$$\textsc{Out}\quad (i : \mathtt{out}(u).P \uplus \mathcal{P}; \phi; i) \xrightarrow{\mathtt{out}(u)} (i : P \uplus \mathcal{P}; \phi \uplus \{\mathsf{w} \mapsto u\}; i) \quad \text{with } \mathsf{w} \in \mathcal{W} \text{ fresh}$$

$$\textsc{Let}\quad (i : \mathtt{let}\ x = v\ \mathtt{in}\ P \uplus \mathcal{P}; \phi; i) \xRightarrow{\tau} (i : P\{x \mapsto v\downarrow\} \uplus \mathcal{P}; \phi; i) \quad \text{when } v\downarrow \in \mathcal{M}_\Sigma.$$

$$\textsc{New}\quad (i : \mathtt{new}\ n.P \uplus \mathcal{P}; \phi; i) \xRightarrow{\tau} (i : P\{n \mapsto n'\} \uplus \mathcal{P}; \phi; i) \quad \text{with } n' \in \mathcal{N} \text{ fresh.}$$

$$\textsc{Rep}\quad (i : !P \uplus \mathcal{P}; \phi; i) \xRightarrow{\tau} (i : P \uplus (i : !P) \uplus \mathcal{P}; \phi; i)$$

$$\textsc{Move}\quad (\mathcal{P}; \phi; i) \xrightarrow{\mathsf{phase}\ i'} (\mathcal{P}; \phi; i') \quad \text{with } i' > i.$$

$$\textsc{Phase}\quad (i : i' : P \uplus \mathcal{P}; \phi; i) \xRightarrow{\tau} (i' : P \uplus \mathcal{P}; \phi; i)$$

Fig. 4: Semantics for processes with phases

6 Getting rid of the topology

We have reduced the topology but we have still to take into account it when analysing the protocol preventing us to use automatic verification tool dedicated to traditional security protocol such as ProVerif [9]. In this section, we will explain how to get rid of the resulting topology and obtain interesting results on timed protocols relying on the notion of phases that is offered to us in ProVerif.

6.1 ProVerif in a nutshell

We consider a subset of the ProVerif calculus defined as follows:

$$P, Q := 0 \mid \mathtt{in}(x).P \mid \mathtt{let}\ x = v\ \mathtt{in}\ P \mid \mathtt{new}\ n.P \mid \mathtt{out}(u).P \mid i : P \mid !P$$

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

The semantics is similar to the one introduced earlier, and formally defined through a relation, denoted $\Rightarrow$, over configurations. A configuration is a tuple $(\mathcal{P}; \phi; i)$ where $\mathcal{P}$ is a multiset of processes (as given by the grammar), ϕ is a frame as usual (with no decoration on the arrow), and $i \in \mathbb{N}$ is an integer that indicates the current phase. Intuitively, the process $!P$ executes P an arbitrary number of times (in parallel), and only processes in the current phase are allowed to evolve. We often write P instead of $0 : P$.

6.2 Our transformation

Given a topology $\mathcal{T}$ (typically one in Figure 3), a protocol $\mathcal{P}_{\mathsf{prox}}$, a role V_0, and a template $\mathcal{I}_0$, we build a configuration $(\mathcal{P}; \phi; 0)$ on which the security analysis could be done using ProVerif. From now on, we assume that $V_0(v_0, p_0)$ only contains one block of the form $\mathtt{reset}.\mathtt{out}(m).\mathtt{in}^{<t}(x)$, i.e. it is of the form:

$$\mathsf{block}_1\ .\ \mathtt{reset}\ .\ \mathtt{out}(m)\ .\ \mathtt{in}^{<t}(x)\ .\ \mathsf{block}_2\ .\ \mathsf{end}(v_0, p_0)$$

where $\mathsf{block_i}$ is a sequence of actions (only simple inputs, outputs, let, and new instructions are allowed).

The main idea is to use phase 1 to represent the critical phase. Such a phase starts when V_0 performs its `reset` instruction, and ends when V_0 performs its $\mathtt{in}^{<t}(x)$ instruction. During this critical phase, only participants that are close enough to V_0 can manipulate messages outputted in this critical phase. The other ones are intuitively too far. Therefore, we mainly consider two transformations, namely $\mathcal{F}^{<}$ and $\mathcal{F}^{\geq}$, whose purposes are to transform a parametrised role of our process algebra given in Section 3.1 (with no `reset` instruction and no guarded input) into a process in the ProVerif calculus.

- *transformation $\mathcal{F}^{<}$:* this transformation introduces the phase instructions with $i = 0$, 1 and 2 considering all the possible ways of splitting the role into three phases (0, 1, and 2). Each phase instruction is placed before an `in` instruction. Such a slicing is actually sufficient for our purposes.
- *transformation $\mathcal{F}^{\geq}$:* this transformation does the same but we forbid the use of the instruction phase 1, jumping directly from phase 0 to phase 2.

The configuration, denoted $\mathcal{F}(\mathcal{T}, \overline{\mathcal{P}_{\mathsf{prox}}}, V_0, \mathcal{I}_0, t_0)$, is the tuple $(\mathcal{P}; \phi; 0)$ where ϕ is such that $img(\phi) = \bigcup_{a \in \mathcal{M}_0} \mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_0)$, and $\mathcal{P}$ is the multiset that contains the following processes (assuming in addition that $a_0 \neq v_0$ when considering the simple distance fraud scenario):
- block_1 . $1 : \mathtt{out}(m)$. $\mathtt{in}^{<t}(x)$. $2 : \mathsf{block}_2$. $\mathsf{end}(v_0, p_0)$;
- $!R(a_0, .., a_n)$ when $R(z_0, .., z_n) \in \mathcal{F}^{<}(\overline{\mathcal{P}_{\mathsf{prox}}})$, $a_0, .., a_n \in \mathcal{A}_0$, $\mathsf{Dist}_{\mathcal{T}}(v_0, a_0) < t_0$;
- $!R(a_0, .., a_n)$ when $R(z_0, .., z_n) \in \mathcal{F}^{\geq}(\overline{\mathcal{P}_{\mathsf{prox}}})$, $a_0, .., a_n \in \mathcal{A}_0$, $\mathsf{Dist}_{\mathcal{T}}(v_0, a_0) \geq t_0$;

Relying on Proposition 2, we are able to establish the following result that justifies the transformation presented above.

Proposition 3. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology, $\mathcal{P}_{\mathsf{prox}}$ a protocol, $t_0 \in \mathbb{R}_+$, $\mathcal{I}_0$ a template, and $V_0(z_0, z_1)$ a parametrised process of the form:*

$$\mathsf{block}_1 \text{ . } \mathtt{reset} \text{ . } \mathtt{out}(m) \text{ . } \mathtt{in}^{<t}(x) \text{ . } \mathsf{block}_2 \text{ . } \mathsf{end}(z_0, z_1) \quad \textit{with } t \leq 2 \times t_0$$

Let K_0 be a valid initial configuration for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$. If K_0 admits an attack w.r.t. t_0-proximity in $\mathcal{T}$, then we have that:

$$\mathcal{F}(\mathcal{T}_0, \overline{\mathcal{P}_{\mathsf{prox}}}, V_0, \mathcal{I}_0, t_0) \overset{\mathsf{tr}}{\Longrightarrow} (\{2 : \mathsf{end}(v_0, p_0)\} \uplus \mathcal{P}; \phi; 2).$$

Moreover, in case there is no $a \in \mathcal{M}_0$ such that $\mathsf{Dist}_{\mathcal{T}_0}(a, v_0) < t_0$, we have that for any $\mathtt{in}(u)$ occurring in tr during phase 1, the underlying recipe R is either of the form w, or only uses handles ouputted in phase 0.

We will see in the following section how this result can be used to turn any attack scenario corresponding to a distance fraud (resp. mafia fraud) into a reachability property, namely the reachability of the event `end`. Regarding distance hijacking, in order to avoid false attacks, we will exploit the additional condition stated at the end of Proposition 3. In particular, a slight modification of the ProVerif code consisting in discarding some attacker behaviours will give us enough precision to obtain meaningful results.

7 Case studies using ProVerif

7.1 Methodology

Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol for which we want to establish the absence of attacks w.r.t. t_0-proximity. Regardless of the type of the considered attack, thanks to our reduction results (Section 5), we only need to consider a single topology (one depicted in Figure 3). Once down to this single topology, we can apply Proposition 3, and analyse the configuration $(\mathcal{P}; \phi; 0) = \mathcal{F}(\mathcal{T}^{t_0}_{\mathsf{XX}}, \overline{\mathcal{P}_{\mathsf{prox}}}, V_0, \mathcal{I}_0, t_0)$ with $\mathsf{XX} \in \{\mathsf{DF}, \mathsf{MF}, \mathsf{DH}\}$ in ProVerif. If the protocol is proved secure, then $\mathcal{P}_{\mathsf{prox}}$ is resistant to the class of attacks we have considered. Otherwise, the trace returned by ProVerif can be analysed to see if it is executable in our timed semantics, and thus corresponds to a real attack.

Actually, to obtain meaningful results regarding scenarios that only involved honest participants in the neighbourhood of v_0, we have to go one step further. Indeed, the attacker model behind ProVerif allows him to interact with any participant (even those that are far away) with no delay. To avoid these behaviours that are not possible in the rapid phase, we slightly modify the ProVerif code taking advantage of the extra condition stated in Proposition 3. During phase 1, we consider an attacker who is only able to forward messages previously sent, and forged new messages using his knowledge obtained in phase 0. We also note that, due to some optimisations in ProVerif code, we couldn't prevent the attacker from using native tuples in phase 1, therefore, we model tuples using our own function symbols.

7.2 Application to distance bounding protocol

In this section, we apply our methodology to a number of well-known distance bounding protocols. However, they may have been abstracted to be analysed in the symbolic framework. In particular, in symbolic models, it is not possible to reason at the bit-level, and therefore we replace the bit-sized exchanges by a single challenge-response exchange using a fresh nonce (as done in Example 4). Sometimes, we also abstract the answer from the prover relying on an uninterpreted function symbol with relevant arguments. Finally, in order to rely on ProVerif, the xor operator has been abstracted (even if our theoretical development is generic enough to deal with such an operator). In constrast with [25] in which the xor operator is fully abstracted relying on a non interpreted function symbol, we actually model it through the following rewriting rules:

$$(x \oplus y) \oplus x \to y \quad (x \oplus y) \oplus y \to x \quad x \oplus (x \oplus y) \to y \quad y \oplus (x \oplus y) \to x$$

In Figure 5, we present the protocols we have analysed using our methodology (plain arrows denote the rapid exchange phase). The Meadows *et al.* protocol [27] has been studied with two different functions $F(n_V, n_P, P) = (n_V \oplus n_P, P)$ and $F(n_V, n_P, P) = (n_V, n_P \oplus P)$ using the weaken xor operator: the first one has been proved secure, while the second one is vulnerable to an attack. All the results are presented in Table 1. We can note that due to the abstractions we

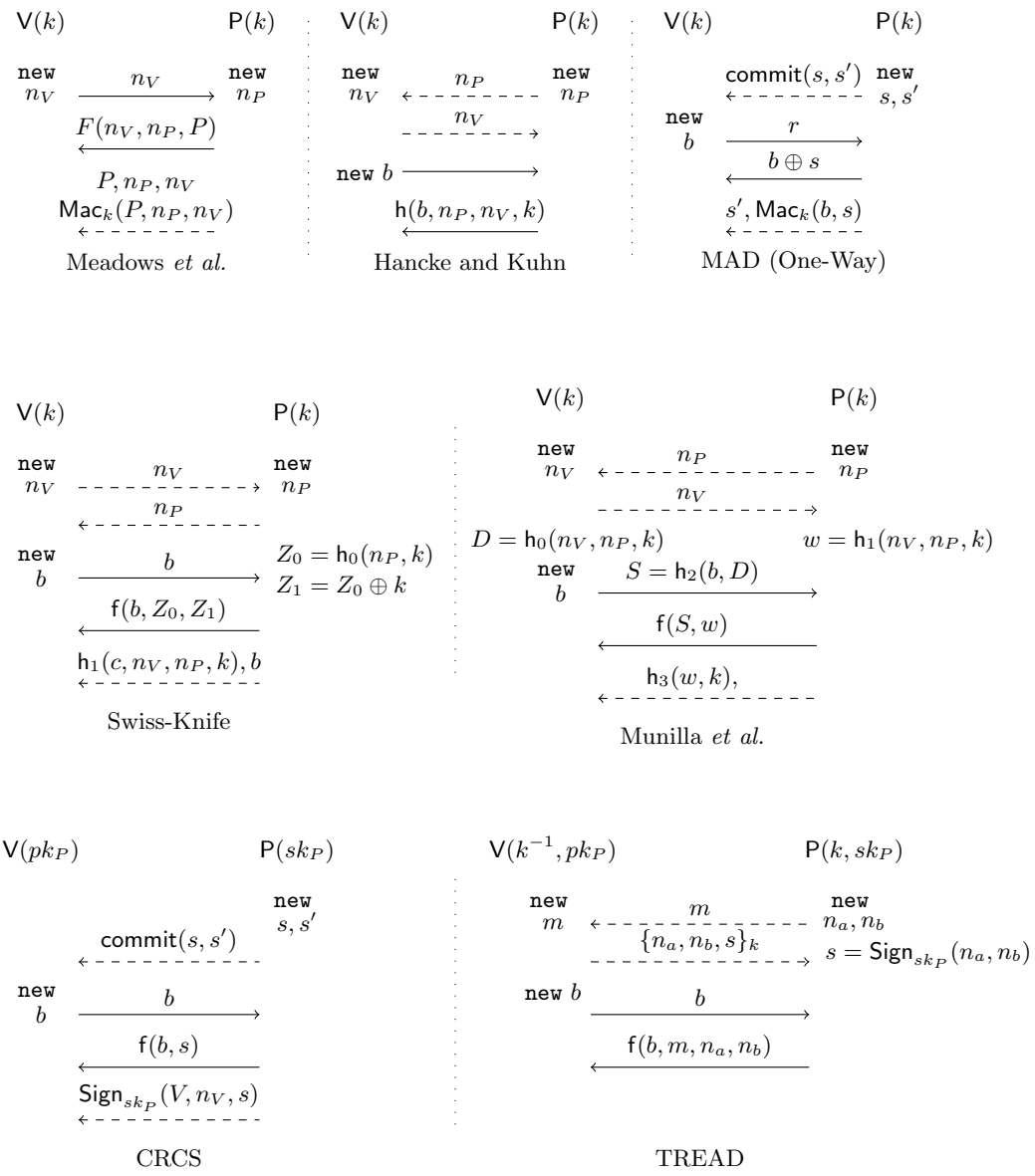


Fig. 5: Description of our case studies

have performed, some protocols, namely Hanckeand Kuhn, Tree-based, Poulidor, and Uniform, become equivalent.

The results are consistent with the ones obtained in [25, 20]. Moreover, our method (with a weakened xor operator) enables us to retrieve automatically the distance hijacking attacks already known on the Meadows *et al.* protocol. In

Protocols	sDF	DF	MF	DH
Brands and Chaum [12]	✓	×	✓	×
MAD (One-Way) [13]	✓	×	✓	×
Meadows *et al.* $(n_V \oplus n_P, P)$ [27]	✓	✓	✓	✓
Meadows *et al.* $(n_V, n_P \oplus P)$ [27]	✓	×	✓	×
Swiss-Knife [24]	✓	✓	✓	✓
Munilla *et al.* [29]	✓	✓	✓	✓
TREAD-Asymmetric [6]	✓	×	×	×
TREAD-Symmetric [6]	✓	×	✓	×
CRCS [31]	✓	×	✓	×
Hancke and Kuhn [22]	✓	✓	✓	✓
Tree-based [7]	✓	✓	✓	✓
Poulidor [32]	✓	✓	✓	✓
Uniform [26]	✓	✓	✓	✓

Table 1: Results on our case studies (×: attack found, ✓: proved secure) (s)DF = (simple) Distance Fraud, MF = Mafia Fraud, DH = Distance Hijacking

comparison, the methodology proposed by Mauw *et al.* in [25] requires to use a specific model of the protocol to retrieve it (e.g. some rules need to be duplicated to simulate some algebraic properties of the xor operator). Another interesting point is that we succeed in proving resistance against mafia fraud for the first version of the Meadows *et al.* protocol (using our weakened xor) which could not be proved using the framework proposed in [27].

Our analysis has been performed using a 64-bit MacOs HighSierra computer with 16 Go of RAM memory and a processor Intel Core i5-7287U CPU 3.30GHz. ProVerif always answers in less than one seconds, and we may note that all the traces retuned by ProVerif correspond to real traces (no false positive).

7.3 Application to the Paysafe protocol

In addition to standard distance bounding protocol, we studied an EMV payment protocol, PaySafe [16], designed to be resistant against mafia fraud attacks. More generally, contactless payment protocols need to prevent relay attacks where malicious agents would abuse from an honest agent to perform a payment, which is the mafia fraud scenario.

The Paysafe protocol is schematised in Figure 6 where plain arrows represents the rapid exchange phase. During the initialisation phase, the reader and the card exchange some identifiers, while during the authentication exchange, the reader ensures that the card is legitimate using signatures and certificates verifications. The main idea is to send nonces and constants during the rapid exchange phase (and perform verifications later) since this can always be done in a short amount of time, therefore, increasing the accuracy on the proximity property needed to ensure the security of the protocol.

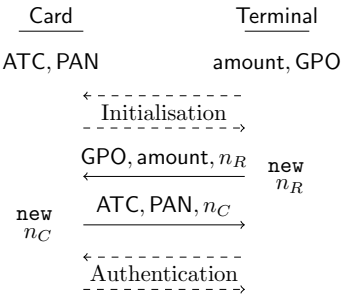


Fig. 6: PaySafe (simplified)

We also considered two other versions of PaySafe, also described in [16], where nonces from the Reader and the Card are removed. Our results, obtained using our methodology with ProVerif, confirmed those presented in [16]. One would note that their methodology and ours, especially when it comes down to the use of ProVerif, are quite similar but we would like to emphasise the fact that our use of ProVerif is a consequence of our formal development. This protocol has also been verified in [25]. The authors of [25] reported a distance fraud attack which is not relevant in this context. Their methodology does not allow them to restrict their analysis to the case of mafia fraud scenario. In contrast, our methodology is flexible enough and thus more suitable in this context.

8 Conclusion

Regarding physical proximity we have shown several reduction results: if there is an attack on an arbitrary topology then there is an attack on a simple one having at most four nodes. Relying on these reduction results, we have shown how to use ProVerif to analyse several protocols provided they make use of primitives supported by the tool. Moreover, our methodology is flexible enough to draw meaningful conclusion on each class of attacks (distance fraud, hijacking attack, mafia fraud).

Our model suffers from some restrictions that prevents it to be used to analyse some scenarios. For instance, we do not consider the notion of terrorist fraud since this would require to consider dishonest participants who only share a part of their knowledge. Another possible extension would be to take also into account the fact that computing messages takes time as it was done *e.g.* in [15], or to consider channels having different speeds. To go even further, quantitative analysis relying on probabilistic models such as those developed in [14] would be beneficial.

References

1. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th Symposium on Principles of Programming Languages (POPL'01)*, pages 104–115. ACM Press, 2001.
2. A. Armando, R. Carbone, L. Compagna, J. Cuéllar, and M. L. Tobarra. Formal analysis of SAML 2.0 web browser single sign-on: breaking the SAML-based single sign-on for Google apps. In *Proc. 6th ACM Workshop on Formal Methods in Security Engineering (FMSE'08)*, pages 1–10. ACM, 2008.
3. A. Armando et al. The AVANTSSAR platform for the automated validation of trust and security of service-oriented architectures. In *Proc. 18th International*

Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'12), volume 7214, pages 267–282. Springer, 2012.
4. G. Avoine, M. Bingol, I. Boureanu, S. Capkun, G. Hancke, S. Kardas, C. Kim, C. Lauradoux, B. Martin, J. Munilla, et al. Security of distance-bounding: A survey. *ACM Computing Surveys*, 2017.
5. G. Avoine, M. A. Bingöl, S. Kardaş, C. Lauradoux, and B. Martin. A framework for analyzing RFID distance bounding protocols. *Journal of Computer Security*, 19(2):289–317, 2011.
6. 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 *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 800–814. ACM, 2017.
7. G. Avoine and A. Tchamkerten. An efficient distance bounding rfid authentication protocol: balancing false-acceptance rate and memory requirement. In *International Conference on Information Security*, pages 250–261. Springer, 2009.
8. 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.
9. 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.
10. 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.
11. I. Boureanu, A. Mitrokotsa, and S. Vaudenay. Practical and provably secure distance-bounding. *Journal of Computer Security*, 23(2):229–257, 2015.
12. S. Brands and D. Chaum. Distance-bounding protocols. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 344–359. Springer, 1993.
13. 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.
14. R. Chadha, A. P. Sistla, and M. Viswanathan. Verification of randomized security protocols. In *Proc. 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS'17)*, pages 1–12. IEEE Computer Society, 2017.
15. V. Cheval and V. Cortier. Timing attacks in security protocols: symbolic framework and proof techniques. In *Proc. 4th Conference on Principles of Security and Trust (POST'15)*, volume 9036 of *LNCS*, pages 280–299. Springer, Apr. 2015.
16. 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 *Lecture Notes in Computer Science*, pages 189–206. Springer, 2015.
17. H. Comon-Lundh and V. Cortier. Security properties: two agents are sufficient. *Programming Languages and Systems*, pages 99–113, 2003.
18. V. Cortier, J. Degrieck, and S. Delaune. Analysing routing protocols: four nodes topologies are sufficient. In *International Conference on Principles of Security and Trust*, pages 30–50. Springer, 2012.
19. V. Cortier and B. Smyth. Attacking and fixing Helios: An analysis of ballot secrecy. *Journal of Computer Security*, 21(1):89–148, 2013.
20. C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *Proc. IEEE Symposium on Security and Privacy (S&P'12)*, pages 113–127. IEEE, 2012.

21. D. Dolev and A. C. Yao. On the security of public key protocols. In *Proc. 22nd Symposium on Foundations of Computer Science (FCS'81)*, pages 350–357. IEEE Computer Society Press, 1981.
22. 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.
23. M. Kanovich, T. B. Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Towards timed models for cyber-physical security protocols. In *Joint Workshop on Foundations of Computer Security and Formal and Computational Cryptography*, 2014.
24. C. H. Kim, G. Avoine, F. Koeune, F.-X. Standaert, and O. Pereira. The swiss-knife RFID distance bounding protocol. In *International Conference on Information Security and Cryptology*, pages 98–115. Springer, 2008.
25. S. Mauw, Z. Smith, J. Toro-Pozo, and R. Trujillo-Rasua. Distance-bounding protocols: Verification without time and location. In *2018 IEEE Symposium on Security and Privacy (S&P)*, volume 00, pages 152–169.
26. S. Mauw, J. Toro-Pozo, and R. Trujillo-Rasua. A class of precomputation-based distance-bounding protocols. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on*, pages 97–111. IEEE, 2016.
27. 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.
28. 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.
29. 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.
30. 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.
31. K. B. Rasmussen and S. Capkun. Realization of rf distance bounding. In *USENIX Security Symposium*, pages 389–402, 2010.
32. R. Trujillo-Rasua, B. Martin, and G. Avoine. The poulidor distance-bounding protocol. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues*, pages 239–257. Springer, 2010.

A Proof of Proposition 1

Proposition 1 is a direct consequence of Proposition 4 stated and proved below. Indeed, it is easy to see that when K is a valid initial configuration then K_0 is also a valid initial configuration, and K_0 is actually a zero-configuration. Moreover, in case the execution $K \to_{\mathcal{T}}^* K'$ under study is an attack trace, then the resulting execution $K_0 = (\mathcal{P}_0; \Phi_0; 0) \to_{\mathcal{T}}^* K_0'$ is also an attack trace.

Proposition 4. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ be a topology and $K = (\mathcal{P}; \Phi; t)$ be a configuration such that any guarded input in $\mathcal{P}$ is preceded by a reset. Let $K' = (\mathcal{P}'; \Phi'; t')$ be a configuration such that $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'$.*

Let $\mathcal{P}_0 = \{ \lfloor P \rfloor_a^0 \mid \lfloor P \rfloor_a^{t_a} \in \mathcal{P}\}$, and $\Phi_0 = \{\mathsf{w} \xrightarrow{a,0} u \mid \mathsf{w} \xrightarrow{a,t_a} u \in \Phi\}$. We have that $K_0 = (\mathcal{P}_0; \Phi_0; 0) \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K_0'$ for some $K_0' = (\mathcal{P}_0'; \Phi_0'; t_0')$ such that:

1. $\{(P, a) \mid \lfloor P \rfloor_a^- \in \mathcal{P}'\} = \{(P, a) \mid \lfloor P \rfloor_a^- \in \mathcal{P}_0'\}$;
2. $\{(\mathsf{w}, a, u) \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi'\} = \{(\mathsf{w}, a, u) \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi_0'\}$.

Proof. We prove the result together with the additional properties stated below by induction on the length of the derivation $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K' = (\mathcal{P}'; \Phi \uplus \Psi'; t')$.

(i) if $\lfloor P \rfloor_a^{t_a} \in \mathcal{P}'$ and P contains a guarded input that is not preceded by a reset then $\lfloor P \rfloor_a^{t_a} \in \mathcal{P}_0'$ (with the same value for t_a);
(ii) $t_0' = t' - t + \delta$ where $\delta = \max(\{\mathsf{Dist}_{\mathcal{T}}(a, b) \mid a, b \in \mathcal{A}_0\} \cup \{t\})$;
(iii) $\Phi_0' = \Phi_0 \uplus \mathsf{Shift}(\Psi', \delta - t)$.

Base case: In such a case, we have that $K' = K$, and thus $t' = t$, $\mathcal{P}' = \mathcal{P}$, $\Phi' = \Phi$, and $\Psi' = \emptyset$. Let $K_0' = (\mathcal{P}_0; \Phi_0; \delta)$ where $\delta = \max(\{\mathsf{Dist}_{\mathcal{T}}(a, b) \mid a, b \in \mathcal{A}_0\} \cup \{t\})$. We have that $K_0 \to_{\mathcal{T}} K_0'$ using the TIM rule. Note that item (i) is satisfied since by hypothesis in $\mathcal{P}'$, any guarded input is preceded by a reset. Regarding (ii), we have that $t_0' = \delta = t' - t + \delta$ since $t' = t$. Since $\Phi_0' = \Phi_0$, and $\Psi' = \emptyset$, item (iii) is also satisfied.

Induction step: In such a case, we have that $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'' \xrightarrow{a,\alpha}_{\mathcal{T}} K'$ with $K'' = (\mathcal{P}''; \Phi''; t'')$ and $\Phi'' = \Phi \uplus \Psi''$. By induction hypothesis, we know that there exists $K_0'' = (\mathcal{P}_0''; \Phi_0''; t_0'')$ such that $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K_0''$ with:

1. $\{(P, a) \mid \lfloor P \rfloor_a^- \in \mathcal{P}''\} = \{(P, a) \mid \lfloor P \rfloor_a^- \in \mathcal{P}_0''\}$;
2. $\{(\mathsf{w}, a, u) \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi''\} = \{(\mathsf{w}, a, u) \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi_0''\}$.

We have also that:

(i) if $\lfloor P \rfloor_a^{t_a} \in \mathcal{P}''$ and P contains a guarded input that is not preceded by a reset then $\lfloor P \rfloor_a^{t_a} \in \mathcal{P}_0''$;
(ii) $t_0'' = t'' - t + \delta$ where $\delta = \max(\{\mathsf{Dist}_{\mathcal{T}}(a, b) \mid a, b \in \mathcal{A}_0\} \cup \{t\})$;
(iii) $\Phi_0'' = \Phi_0 \uplus \mathsf{Shift}(\Psi'', \delta - t)$.

We have to establish that there exists $K_0' = (\mathcal{P}_0'; \Phi_0'; t_0')$ that satisfies the five properties stated above and such that $K_0'' \xrightarrow{a,\alpha}_{\mathcal{T}} K_0'$. We consider the rule involved in $K'' \xrightarrow{a,\alpha}_{\mathcal{T}} K'$ and we show that the same rule can be applied on K_0'', and allows one to get K_0' with the five properties stated above.

Case TIM rule with δ_0. In such a case, we have that $K' = (\mathcal{P}''; \Phi''; t'' + \delta_0)$, and we apply the same rule with the delay δ_0 on K_0''. We obtain $K_0' = (\mathcal{P}_0''; \Phi_0''; t_0'' + \delta_0)$, and we easily check that all the properties are satisfied.

Case OUT rule. In such a case, $K' = (\mathcal{P}'; \Phi'' \uplus \{\mathsf{w} \xrightarrow{a,t''} u\}; t'')$ (for some $\mathcal{P}'$, w, a, and u). We apply the same rule on K_0'', and obtain $K_0' = (\mathcal{P}_0'; \Phi_0'' \uplus \{\mathsf{w} \xrightarrow{a,t_0''} u\}; t_0'')$ (for some $\mathcal{P}_0'$). We have that items 1 and 2 are clearly satisfied as well as item (i). Now, regarding item (ii), we have that $t_0' = t_0''$ and $t' = t''$. Therefore, we conclude that $t_0' = t' - t + \delta$ thanks to our induction hypothesis. Now, to establish that $\Phi_0' = \Phi_0 \uplus \mathsf{Shift}(\Psi', \delta - t)$, relying on our iduction hypothesis, it only remains to show that $t_0'' = t'' + (\delta - t)$ (this is item *ii*).

Case LET and NEW rules. In such a case, $K' = (\mathcal{P}'; \Phi''; t'')$ (for some $\mathcal{P}'$), and we apply that same rule on K_0'', and obtain $K_0' = (\mathcal{P}_0'; \Phi_0''; t_0'')$. We conclude easily (for each property) relying on our induction hypothesis.

Case RST rule. In such a case, $K' = (\mathcal{P}'; \Phi''; t'')$ (for some $\mathcal{P}'$), and we know that $\mathcal{P}'' = \{ \lfloor \mathtt{reset}.P \rfloor_a^{t^a} \} \cup \mathcal{Q}'$, and $\mathcal{P}' = \{ \lfloor P \rfloor_a^0 \} \cup \mathcal{Q}'$ for some P, a, t^a and $\mathcal{Q}'$. We apply the same rule on K_0'', and obtain $K_0' = (\mathcal{P}_0'; \Phi_0''; t_0'')$, and we know that $\mathcal{P}_0'' = \{ \lfloor \mathtt{reset}.P \rfloor_a^{t_0^a} \} \cup \mathcal{Q}_0'$, and $\mathcal{P}_0' = \{ \lfloor P \rfloor_a^0 \} \cup \mathcal{Q}_0'$ for some t_0^a and $\mathcal{Q}_0'$. Relying on our induction hypothesis, we easily obtain the fact that items 1 and 2 are satisfied. Regarding item (i), we conclude using our induction hypothesis for processes in $\mathcal{Q}_0'$, and the property is satisfied for $\lfloor P \rfloor_a^0$. Regarding items (ii) and (iii), since the global time and the frame have not evolved, we conclude relying on our induction hypothesis.

Case IN rule. In such a case, $K' = (\mathcal{P}'; \Phi''; t'')$ (for some $\mathcal{P}'$), and we apply the same rule on K_0'' to get $K_0' = (\mathcal{P}_0'; \Phi_0''; t_0'')$. The difficult part is to show that the rule can indeed be applied on K_0''. By hypothesis, we know that $K'' \xrightarrow{a,\mathtt{in}^\star(u)}_{\mathcal{T}} K'$, and thus $\mathcal{P}'' = \lfloor \mathtt{in}^\star(x).P \rfloor_a^{t^a} \cup \mathcal{Q}$ for some x, a, t^a and $\mathcal{Q}$, and we know that there exists $b \in \mathcal{A}_0$ and $t^b \in \mathbb{R}_+$ such that $t^b < t'' - \mathsf{Dist}_{\mathcal{T}}(b, a)$ and:

- if $b \in \mathcal{A}_0 \setminus \mathcal{M}_0$ 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 $\mathsf{w} \in \mathit{dom}(\lfloor \Phi'' \rfloor_c^{t^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$.

Moreover, in case $\star$ is $< t_g$ for some t_g, we know in addition that $t^a < t_g$.

We first assume that $\mathtt{in}^\star(u)$ is a simple input (not a guarded one). We show that we can apply on K_0'' the same rule using the same recipe R. The message will be sent by the same agent $b \in \mathcal{A}_0$. The time t_0^b at which the message is $t^b + (\delta - t)$. Note that $t_0^b \geq t^b$ since $\delta - t \geq 0$. We distinguish two cases:

- if $b \in \mathcal{A}_0 \setminus \mathcal{M}_0$ then we know that $u \in \mathit{img}(\lfloor \Phi'' \rfloor_b^{t^b}) = \mathit{img}(\lfloor \Phi \rfloor_b^{t^b}) \cup \mathit{img}(\lfloor \Psi'' \rfloor_b^{t^b})$. Actually, we have that $\mathit{img}(\lfloor \Phi \rfloor_b^{t^b}) \subseteq \mathit{img}(\lfloor \Phi_0 \rfloor_b^{t_0^b})$ since $t_0^b \geq 0$

and by definition $\Phi_0 = \{\mathsf{w} \xrightarrow{a,0} u \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi\}$. We have also that $img(\lfloor \Psi'' \rfloor_b^{t^b}) \subseteq img(\lfloor \Psi_0'' \rfloor_b^{t_0^b})$ since $\Psi_0'' = \mathsf{Shift}(\Psi'', \delta - t)$, and $t_0^b = t^b + (\delta - t)$.

– if $b \in \mathcal{M}_0$ then let $\mathsf{w} \in vars(R)$. By hypothesis, we know that there exists $c \in \mathcal{A}_0$ such that $\mathsf{w} \in dom(\lfloor \Phi'' \rfloor_c^{t^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$. To conclude, it is sufficient to show that $\mathsf{w} \in dom(\lfloor \Phi_0'' \rfloor_c^{t_0^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$. Again, we distinguish two cases. In case $\mathsf{w} \in dom(\lfloor \Phi \rfloor_c^{t^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$, then $\mathsf{w} \in dom(\lfloor \Phi_0 \rfloor_c^{t_0^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$ since $t_0^b \geq 0$ and by definition $\Phi_0 = \{\mathsf{w} \xrightarrow{a,0} u \mid \mathsf{w} \xrightarrow{a,-} u \in \Phi\}$. In case $\mathsf{w} \in dom(\lfloor \Psi'' \rfloor_c^{t^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$, then $\mathsf{w} \in dom(\lfloor \Psi_0'' \rfloor_c^{t_0^b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$ since $\Psi_0'' = \mathsf{Shift}(\Psi'', \delta - t)$, and $t_0^b = t^b + (\delta - t)$.

In case $\star$ is $< t_g$, by hypothesis we know that $t^a < t_g$, and thanks to item (i), we know that $\mathcal{P}_0'' = \lfloor \mathtt{in}^\star(x).P \rfloor_a^{t^a} \cup \mathcal{Q}_0$. This allows us to conclude. □

B Proof of Theorem 1 (Distance Fraud)

We show this result in two main steps:

1. We move the agents (keeping p_0 far away from v_0) in a way that will only shorten the distance between any two of them.
2. We rename agents preserving their location (Lemma 1), and we show that the resulting initial configuration is still a valid one.

Lemma 1. *Let $K = (\mathcal{P}; \Phi; t)$ be a configuration on $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ and $\rho : \mathcal{A} \to \mathcal{A}_0$ be a renaming such that $\mathsf{Loc}_0(\rho(a)) = \mathsf{Loc}_0(a)$ for any $a \in \mathcal{A}_0$, and if $a \in \mathcal{M}_0$ then $\rho(a) \in \mathcal{M}_0$. Let $K' = (\mathcal{P}'; \Phi'; t')$ be a configuration such that $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'$. We have that:*

$$K\rho \xrightarrow{\mathsf{tr}\rho}_{\mathcal{T}} K'\rho$$

Proof. We show this result by induction on the length of $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'$. The base case, i.e. tr is the empty trace, is trivial. To conclude, it is sufficient to show that $K_1 \xrightarrow{a,\alpha}_{\mathcal{T}} K_2$ with K_1 a configuration built on $\mathcal{T}$ implies that:

$$K_2 \text{ is a configuration built on } \mathcal{T}, \text{ and } K_1\rho \xrightarrow{\rho(a),\rho(\alpha)}_{\mathcal{T}} K_2\rho.$$

K_1 only involves processes located at $a \in \mathcal{A}_0$, and therefore actions along the derivation are only executed by agents in $\mathcal{A}_0$ whose locations remain unchanged by ρ. We consider each rule of the semantics one by one. The only rules that are not trivial are the rules LET and IN.

Case of the rule LET. In such a case, we have that $K_1 = (\mathcal{P}_1; \Phi_1; t_1)$ and $K_2 = (\mathcal{P}_2; \Phi_2; t_2)$ with $\mathcal{P}_1 = \lfloor \mathtt{let}\ x = u\ \mathtt{in}\ P \rfloor_a^{t_a} \uplus \mathcal{P}$, $\mathcal{P}_2 = \lfloor P\{x \mapsto u\downarrow\} \rfloor_a^{t_a} \uplus \mathcal{P}$, $\Phi_2 = \Phi_1$, and $t_2 = t_1$. We know that $u\downarrow \in \mathcal{T}(\Sigma_c^+, \mathcal{N} \uplus \mathcal{A})$, and therefore we have that $(u\rho)\downarrow \in \mathcal{T}(\Sigma_c^+, \mathcal{N} \uplus \mathcal{A})$ by applying the same rewriting rule at each step. This allows us to apply the rule LET and to obtain the expected result.

Case of the rule IN. In such a case, we have that $K_1 = (\mathcal{P}_1; \Phi_1; t_1)$ and $K_2 = (\mathcal{P}_2; \Phi_2; t_2)$ with $\mathcal{P}_1 = \lfloor \mathtt{in}^\star(x).P \rfloor_a^{t_a} \uplus \mathcal{P}$, $\mathcal{P}_2 = \lfloor P\{x \mapsto u\} \rfloor_a^{t_a} \uplus \mathcal{P}$, $\Phi_2 = \Phi_1$, and $t_2 = t_1$. We know that there exists $b \in \mathcal{A}_0$, $t_b \in \mathbb{R}^+$ such that $t_b \leq t_1 - \mathsf{Dist}_{\mathcal{T}}(b, a)$, and a recipe R such that $u = R\Phi_1\downarrow$, and for all $\mathsf{w} \in vars(R)$ there exists $c \in \mathcal{A}_0$ such that $w \in dom(\lfloor \Phi_1 \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$. Moreover, when $b \in \mathcal{A}_0 \setminus \mathcal{M}_0$, we know that $R \in \mathcal{W}$. To conclude that the rule IN can be applied, we simply have to show that $R(\Phi\rho)\downarrow = u\rho$. Note that the renaming ρ keeps the locations of the agents in $\mathcal{A}_0$ unchanged, and therefore the conditions about distance are still satisfied. We have that $R(\Phi\rho)\downarrow = (R\Phi)\rho\downarrow$ since $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, \mathcal{W})$. We know that $(R\Phi)\downarrow = u$, and therefore we have that $(R\Phi)\rho\downarrow = u\rho$ by applying the same rewriting rule at each step. Note that $u\rho$ does not contain any destructor symbol and is thus in normal form. To conclude, it remains to ensure that when $\rho(b) \in \mathcal{A}_0 \setminus \mathcal{M}_0$, then $R \in \mathcal{W}$. Actually, we know that if $\rho(b) \notin \mathcal{M}_0$ then $b \notin \mathcal{M}_0$ by hypothesis, and this allows us to conclude. □

Theorem 1. *Let $\mathcal{P}_{\mathsf{prox}}$ be a protocol, $V_0(z_0, z_1)$ be a parametrised role containing the special event $\mathsf{end}(z_0, z_1)$, and $\mathcal{I}_0$ be a template. We have that $\mathcal{P}_{\mathsf{prox}}$ admits a distance fraud attack w.r.t. t_0-proximity, if and only if, there is an attack against t_0-proximity in the topology $\mathcal{T}^{t_0}_{\mathsf{DF}}$.*

Proof. Since $\mathcal{T}^0_{\mathsf{DF}} \in \mathcal{C}_{\mathsf{DF}}$, one direction is trivial. We consider the other one. Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0) \in \mathcal{C}_{\mathsf{DF}}$ and $K_0 = (\mathcal{P}_0; \Phi_0; t_{\mathsf{init}})$ be a valid initial configuration for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$. We have that:

$$K_0 \to^*_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_v} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0$$

We consider $\mathcal{T}' = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}', v_0, p_0)$ such that $\mathsf{Loc}'(v_0) = \mathsf{Loc}_0(v_0)$, and:

$$\mathsf{Loc}'(a) = \mathsf{Loc}'(b) \text{ for any } a, b \in \mathcal{A}_0 \setminus \{v_0\} \text{ with } \mathsf{Dist}_{\mathcal{T}'}(v_0, p_0) = t_0.$$

Note that, since $\mathsf{Dist}_{\mathcal{T}}(v_0, a) \geq t_0$ for any $a \in \mathcal{A}_0 \setminus \{v_0, p_0\}$ by definition of $\mathcal{T} \in \mathcal{C}_{\mathsf{DF}}$, we have that $\mathsf{Dist}_{\mathcal{T}'}(a, b) \leq \mathsf{Dist}_{\mathcal{T}}(a, b)$ for any $a, b \in \mathcal{A}_0$. Thus:

$$K_0 \to^*_{\mathcal{T}'} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_v} \uplus \mathcal{P}; \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}'}(v_0, p_0) \geq t_0$$

Then, we consider the renaming $\rho : \mathcal{A} \to \mathcal{A}_0$ such that $\rho(v_0) = v_0$, and $\rho(a) = p_0$ otherwise. We have that $\mathsf{Loc}'(\rho(a)) = \mathsf{Loc}'(a)$ for any $a \in \mathcal{A}_0$, and in case $a \in \mathcal{M}_0$, we know that $a \neq v_0$, and thus we have that $\rho(a) = p_0 \in \mathcal{M}_0$ (since $\mathcal{T} \in \mathcal{C}_{\mathsf{DF}}$). Thus, we can apply Lemma 1, we obtain that:

$$K_0\rho \to^*_{\mathcal{T}'} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_v} \uplus \mathcal{P}\rho; \Phi\rho; t) \text{ with } \mathsf{Dist}_{\mathcal{T}'}(v_0, p_0) \geq t_0.$$

Since $\mathcal{T}'$ and $\mathcal{T}^0_{\mathsf{DF}}$ coincide on $img(\rho) = \{p_0, v_0\}$, we have that:

$$K_0\rho \to^*_{\mathcal{T}^0_{\mathsf{DF}}} (\lfloor \mathsf{end}(v, p) \rfloor_{v}^{t_v} \uplus \mathcal{P}\rho; \Phi\rho; t) \text{ with } \mathsf{Dist}_{\mathcal{T}^0_{\mathsf{DF}}}(v_0, p_0) \geq t_0.$$

To conlude, it remains to show that $K_0\rho$ is a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$, and V_0 w.r.t. $\mathcal{T}^0_{\mathsf{DF}}$ and $\mathcal{I}_0$. We have to prove that $img(\lfloor \Phi\rho \rfloor_{v_0}^{t_{\mathsf{init}}}) = \emptyset$, and $img(\lfloor \Phi\rho \rfloor_{p_0}^{t_{\mathsf{init}}}) =$

$\mathsf{Knows}(\mathcal{I}_0, p_0, \{v_0, p_0\})$. We have that:

$$\begin{aligned} img(\lfloor \Phi_0\rho \rfloor_{p_0}^{t_{\mathsf{init}}}) &= \bigcup_{\{a\in\mathcal{A}_0 \mid \rho(a)=p_0\}} (img(\lfloor \Phi_0 \rfloor_a^{t_{\mathsf{init}}})\rho \\ &= \bigcup_{\{a\in\mathcal{A}_0 \mid \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_0)\rho \\ &= \bigcup_{\{a\in\mathcal{A}_0 \mid \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, \rho(a), \rho(\mathcal{A}_0)) \\ &= \bigcup_{\{a\in\mathcal{A}_0 \mid \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, p_0, \{p_0, v_0\}) \end{aligned}$$

Moreover, we have that $img(\lfloor \Phi_0\rho \rfloor_{v_0}^{t_{\mathsf{init}}}) = img(\lfloor \Phi_0 \rfloor_{v_0}^{t_{\mathsf{init}}})\rho = \emptyset$. This allows us to conclude. □

C Proof of Theorem 2 (Mafia Fraud)

We show this result in three main steps:

1. We reduce the number of active agents and show that it is sufficient to keep only 2 active agents, namely v_0 and p_0. More precisely we first transform honest agents into dishonest ones (Lemma 2). Then, relying on our executability hypothesis, we discard processes executed by malicious agents (Proposition 5).
2. We control the number of attackers by placing them ideally, i.e. right next each active agent (Proposition 6).
3. We rename agents preserving their location (Lemma 1), and we show that the resulting initial configuration is still a valid one.

Reducing the number of active agents.

Lemma 2. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology, and K_0 be a configuration built on $\mathcal{T}$. Let $\mathcal{H}_0 \subseteq \mathcal{A}_0 \setminus \mathcal{M}_0$. Let K be a configuration such that $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K$. We have that $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}'} K$ where $\mathcal{T}' = (\mathcal{A}_0, \mathcal{M}_0 \cup \mathcal{H}_0, \mathsf{Loc}_0, v_0, p_0)$.*

Proof. We show this result by induction on the length of the derivation $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K$. The base case, i.e. tr is the empty trace, is trivial. To conclude, it is sufficient to show that:

$$K_1 \xrightarrow{a,\alpha}_{\mathcal{T}} K_2 \text{ implies } K_1 \xrightarrow{a,\alpha}_{\mathcal{T}'} K_2.$$

Let $K_1 = (\mathcal{P}_1; \Phi_1, t_1)$. We consider each rule of the semantics one by one. Actually, the only rule that depends on the status (honest/dishonest) of the agents of the underlying topology is the rule IN. In such a case, we have that $\alpha = \mathtt{in}^\star(u)$ for some u. Moreover, following the notation introduced in Section 3.3, we know that there exist $b \in \mathcal{A}_0$ (the agent responsible of the corresponding output) and $t_b \in \mathbb{R}_+$ (the time at which the output has been triggered). The only interesting case is when $b \in \mathcal{H}_0$, and therefore b is now a malicious agent in the topology $\mathcal{T}'$ whereas b was an honest one in the topology $\mathcal{T}$. Since, the IN rule was triggered in $\mathcal{T}$, we know that $u \in img(\lfloor \Phi_1 \rfloor_b^{t_b})$, and therefore there exists $\mathsf{w} \in dom(\lfloor \Phi_1 \rfloor_b^{t_b})$ such that $\mathsf{w}\Phi_1\downarrow = u$. Actually, it is easy to see that choosing the recipe $R = \mathsf{w}$ (and $c = b$) allows us to conclude. Indeed, we have that $t_b - \mathsf{Dist}_{\mathcal{T}'}(b, b) = t_b$, and therefore we conclude since we have already shown that $\mathsf{w} \in dom(\lfloor \Phi_1 \rfloor_b^{t_b})$. □

Definition 8. *A closed process P without new construction is executable w.r.t. a frame Φ if for any term u (resp. v) occurring in an output or a let there exists a term $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Phi) \cup \mathcal{X})$ such that $u = R\Phi\downarrow$ (resp. $v\downarrow = R\Phi\downarrow$).*

Proposition 5. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology, $K_0 = (\mathcal{P}_0; \Phi_0; t_0)$ be a configuration built on $\mathcal{T}$, and $\mathcal{D}_0 \subseteq \mathcal{M}_0$. We assume that for any $\lfloor P_a \rfloor^{t_a}_a \in \mathcal{P}_0$ with $a \in \mathcal{D}_0$, we have that P_a is executable w.r.t. $\lfloor \Phi_0 \rfloor^{t_0}_a$.*

Let K be a configuration such that $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K = (\mathcal{P}; \Phi_0 \uplus \Phi^+; t)$. We have that $(\overline{\mathcal{P}_0}; \Phi_0, t_0) \xrightarrow{\overline{\mathsf{tr}}}_{\mathcal{T}} (\overline{\mathcal{P}}; \Phi_0 \uplus \overline{\Phi^+}; t)$ where $\overline{\mathcal{P}}$ (resp. $\overline{\Phi^+}$, $\overline{\mathsf{tr}}$) is obtained from $\mathcal{P}$ (resp. Φ^+, tr) by removing processes (resp. frame or trace elements) located in $a \in \mathcal{D}_0$.

Proof. We will show this result assuming that $\mathcal{D}_0$ contains a unique element a_0. The general result can then easily be obtained by a simple induction on the size of $\mathcal{D}_0$. More precisely, we show the following results by induction on the length of $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K = (\mathcal{P}; \Phi_0 \uplus \Phi^+; t)$:

1. for all $\lfloor P \rfloor^{t_a}_{a_0} \in \mathcal{P}$, P is executable w.r.t. $\Psi(t)$;
2. for all $\mathsf{w} \xrightarrow{a_0, t_a} u \in \Phi^+$, there exists $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Psi(t_a)))$ such that $R\Psi(t_a)\downarrow = u$;
3. $(\overline{\mathcal{P}_0}; \Phi_0, t_0) \xrightarrow{\overline{\mathsf{tr}}}_{\mathcal{T}} (\overline{\mathcal{P}}; \Phi_0 \uplus \overline{\Phi^+}; t)$;

where $\Psi(t) = \Phi_0 \uplus \{ \lfloor \Phi^+ \rfloor^{t - \mathsf{Dist}_{\mathcal{T}}(b, a_0)}_b \mid b \neq a_0 \}$.

The base case, i.e. when tr is empty, is trivial. Now, we assume that

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K = (\mathcal{P}; \Phi_0 \uplus \Phi^+; t) \xrightarrow{a, \alpha} K' = (\mathcal{P}'; \Phi_0 \uplus \Phi'; t')$$

and thanks to our induction hypothesis, we know that the three properties above hold on K. We do a case analysis on the rule involved in the last step.

<u>*Rule TIM.*</u> In such a case, we have that $\Phi' = \Phi^+$ and $\mathcal{P}' = \mathcal{P}$. Since $t' \geq t$, we have that $\Psi(t) \subseteq \Psi(t')$, and this allows us to conclude.

<u>*Rule OUT.*</u> Items 1 and 3 are quite obvious. Regarding item 2, the only non trivial case is when $a = a_0$. We have to show that the element added to the frame, namely $\mathsf{w} \xrightarrow{a_0, t} u$ satisfies the expected property. By hypothesis, we know that the process $\mathtt{out}(u).P$ responsible of this output is executable w.r.t. $\Psi(t)$, and thus there exists $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Psi(t)))$ such that $R\Psi(t)\downarrow = u\downarrow$.

<u>*Rule LET.*</u> The only non trivial point is to establish item 1 when $a = a_0$. We have that $\mathcal{P} = \lfloor \mathtt{let}\ x = v\ \mathtt{in}\ P' \rfloor^{t_a}_{a_0} \uplus \mathcal{P}_1$, $\mathcal{P}' = \lfloor P'\{x \mapsto v\downarrow\} \rfloor^{t_a}_{a_0} \uplus \mathcal{P}_1$, $\Phi' = \Phi^+$, and $t' = t$. By hypothesis, we know that the process $\mathtt{let}\ x = v\ \mathtt{in}\ P'$ in $\mathcal{P}$ is executable w.r.t. $\Psi(t)$, thus there exists $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Psi(t)))$ such that $R\Psi(t)\downarrow = v\downarrow$. To prove item 1, we have to show that $P'\{x \mapsto v\downarrow\}$ is executable w.r.t. $\Psi(t') = \Psi(t)$. Let u be a term occurring in an output (resp. a let) in $P'\{x \mapsto v\downarrow\}$. We have that there exists u_0 that occurs in an ouput (resp. a let) in P' such that $u_0\{x \mapsto v\downarrow\} = u$, and by hypothesis, we know that $\mathtt{let}\ x = v\ \mathtt{in}\ P'$

is executable w.r.t. $\Psi(t)$, i.e. there exists $R_0 \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Psi(t)) \cup \mathcal{X})$ such that $u_0{\downarrow} = R_0\Psi(t){\downarrow}$. Let $R' = R_0\{x \mapsto R\}$. We have that

$$R'\Psi(t'){\downarrow} = (R_0\{x \mapsto R\})\Psi(t){\downarrow} = (R_0\Psi(t)\{x \mapsto v{\downarrow}\}){\downarrow} = (u_0\{x \mapsto v{\downarrow}\}){\downarrow} = u{\downarrow}.$$

Note that $u = u_0\{x \mapsto v{\downarrow}\}$ is in normal form when u_0 is is normal form. Thus, no normalisation is needed in case u is a term occurring in an output.

Rule NEW. We know that $a \neq a_0$, and thus the result trivially holds.

Rule RST. In such a case, the result trivially holds.

Rule IN. In case $a = a_0$, the only non trivial point is to establish item 1, and this can be done in a similar way as it was done in Rule LET. Otherwise, i.e. when $a \neq a_0$, the non trivial point is to establish item 3. Let $\alpha = \mathtt{in}^\star(u)$. We have to establish that the IN rule can still be fired with the same value u despite the fact that some frame elements have been removed from Φ^+.
Following the notations introduced in Section 3.3, we know that there exist $b \in \mathcal{A}_0$ and $t_b \in \mathbb{R}_+$ such that $t_b \leq t - \mathsf{Dist}_{\mathcal{T}}(b, a)$ and a recipe R such that $R(\Phi_0 \uplus \Phi^+){\downarrow} = u$ and for all $\mathsf{w} \in vars(R)$ there exists $c \in \mathcal{A}_0$ such that $\mathsf{w} \in dom(\lfloor \Phi_0 \uplus \Phi^+ \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$.

1. If $b \notin \mathcal{M}_0$ then we know that $R = \mathsf{w}$ and we have that $\lfloor \Phi_0 \uplus \Phi^+ \rfloor_b^{t_b} = \lfloor \Phi_0 \uplus \overline{\Phi^+} \rfloor_b^{t_b}$ because $b \neq a_0$. Thus, the rule can be applied.
2. If $b \in \mathcal{M}_0$, then in case $c \neq a_0$, or $c = a_0$ with $\mathsf{w} \in \Phi_0$, then it is straightforward. The interesting case is when $\mathsf{w} \in dom(\lfloor \Phi^+ \rfloor_{a_0}^{t_b - \mathsf{Dist}_{\mathcal{T}}(a_0,b)})$, and we have to reconstruct $\mathsf{w}\Phi^+$ with elements available in $\Phi_0 \uplus \overline{\Phi^+}$. To do so, we apply item 2: we obtain that there exists a time $t_a \leq t_b - \mathsf{Dist}_{\mathcal{T}}(a_0, b)$ and a recipe $R_{\mathsf{w}} \in \mathcal{T}(\Sigma_{\mathsf{pub}}, dom(\Psi(t_a)))$ such that $R_{\mathsf{w}}\Psi(t_a){\downarrow} = \mathsf{w}\Phi^+$. By definition of $\Psi(t_a)$, we have that $\Psi(t_a) = \Phi_0 \uplus \{ \lfloor \Phi^+ \rfloor_c^{t_a - \mathsf{Dist}_{\mathcal{T}}(c,a_0)} \mid c \neq a_0 \}$. Moreover, we know that

$$t_a - \mathsf{Dist}_{\mathcal{T}}(c, a_0) \leq t_b - (\mathsf{Dist}_{\mathcal{T}}(c, a_0) + \mathsf{Dist}_{\mathcal{T}}(a_0, b)) \leq t_b - \mathsf{Dist}_{\mathcal{T}}(c, b).$$

   Thus, we have that $vars(R_{\mathsf{w}}) \subseteq dom(\Phi_0) \uplus dom(\{ \lfloor \Phi^+ \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}}(c,b)} \mid c \neq a_0 \})$. We consider the substitution σ with $dom(\sigma) = dom(\lfloor \Phi^+ \rfloor_{a_0}^{t_b - \mathsf{Dist}_{\mathcal{T}}(a_0,b)})$ and such that $\sigma(\mathsf{w}) = R_{\mathsf{w}}$. We have that $R\sigma \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, dom(\Phi_0 \uplus \overline{\Phi^+}))$ and

$$\begin{aligned} R\sigma(\Phi_0 \uplus \overline{\Phi^+}){\downarrow} &= R(\Phi_0 \uplus \{\mathsf{w} \mapsto R_{\mathsf{w}}(\Phi_0 \uplus \overline{\Phi^+}) \mid \mathsf{w} \in dom(\sigma)\}){\downarrow} \\ &= R(\Phi_0 \uplus \{\mathsf{w} \mapsto \mathsf{w}\Phi^+ \mid \mathsf{w} \in dom(\sigma)\}){\downarrow} \\ &= R(\Phi_0 \uplus \Phi^+){\downarrow} \\ &= u. \end{aligned}$$

   Therefore the IN rule can be applied. □

In order to be able to apply Proposition 5, we will have to clean the process by removing new instructions, and applying a suitable renaming.

Lemma 3. *Let $K = (\mathcal{P}; \Phi; t)$ be a configuration built on $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$, and $\mathcal{D} \subseteq \mathcal{A}_0$. Let $K' = (\mathcal{P}'; \Phi'; t')$ be a configuration such that $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'$, and N be the set of names that have been freshly generated to trigger all the rules NEW executed by agents in $\mathcal{D}$. We have that:*

$$(\tilde{\mathcal{P}}; \Phi; t) \xrightarrow{\mathsf{tr}\rho}_{\mathcal{T}} (\tilde{\mathcal{P}'}\rho; \Phi'\rho; t')$$

where ρ replaces any name from N with $c_0 \in \Sigma_0$, and $\tilde{\mathcal{P}}$ is the process obtained from $\mathcal{P}$ by removing "new" instructions of the form $\mathtt{new}\ n$ and replacing the occurrence of n by c_0 in any process located in $a \in \mathcal{D}$.

Proof. We show this result by induction on the length of $K \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'$. The base case, i.e. when tr is the empty trace, is trivial. To conclude, it is sufficient to show that $(\mathcal{P}_1; \Phi_1; t_1) \xrightarrow{a,\alpha}_{\mathcal{T}} (\mathcal{P}_2; \Phi_2; t_2)$ implies that:

1. either $(\tilde{\mathcal{P}_1}\rho; \Phi_1\rho; t_1) = (\tilde{\mathcal{P}_2}\rho; \Phi_2\rho; t_2)$;
2. or $(\tilde{\mathcal{P}_1}\rho; \Phi_1\rho; t_1) \xrightarrow{a,\alpha\rho}_{\mathcal{T}} (\tilde{\mathcal{P}_2}\rho; \Phi_2\rho; t_2)$

Note that since ρ has no effect on the initial configuration K, this will allow us to conclude. We consider each rule of the semantics one by one.

- Rule TIM: In such a case, the same rule applies.
- Rule OUT: In such a case, we have that $\mathcal{P}_1 = \lfloor \mathtt{out}(u).P \rfloor_a^{t_a} \uplus \mathcal{P}'_0$ and $u\downarrow$ will be added into the frame. Actually the same rule applies on the configuration $(\tilde{\mathcal{P}_1}\rho; \Phi_1\rho; t_1)$ and $u\rho\downarrow = u\downarrow\rho$ will be added into the frame.
- Rule LET: In such a case, we have that $\mathcal{P}_1 = \lfloor \mathtt{let}\ x = u\ \mathtt{in}\ P \rfloor_a^{t_a} \uplus \mathcal{P}'_0$ and $u\downarrow$ is a message. Actually the same rule applies on the configuration $(\tilde{\mathcal{P}_1}\rho; \Phi_1\rho; t_1)$ since $u\rho\downarrow = u\downarrow\rho$ is also a message.
- Rule NEW: In case $a \in \mathcal{D}$, since the freshly generated name n' will be in N, we have that $(\tilde{\mathcal{P}_1}\rho; \Phi_1\rho; t_1) = (\tilde{\mathcal{P}_2}\rho; \Phi_2\rho; t_2)$. Otherwise, the same rule applies.
- Rule RST: The same rule applies.
- Rule IN: In such a case, we have that $\mathcal{P}_1 = \lfloor \mathtt{in}(x).P \rfloor_a^{t_a} \uplus \mathcal{P}'_0$, and we have a recipe R such that $R\Phi_1\downarrow = u$ together with some extra conditions. The same rule applies using the same recipe R. We have that $R(\Phi_1\rho)\downarrow = u\rho$.

This allows us to conclude. □

Controlling the number of intuders.

Given a mapping Loc from a set of agents $\mathcal{A}_0 = \{a_1, \ldots, a_p\}$ to $\mathbb{R}^3$, we define the canonical topology $\mathcal{T}_{\mathsf{Loc}}$ associated to Loc as $(\mathcal{A}_0 \uplus \mathcal{M}_0, \mathcal{M}_0, \mathsf{Loc} \uplus \mathsf{Loc}_0, v_0, p_0)$ where:

- $\mathcal{M}_0 = \{i_1, \ldots, i_p\}$ where $i_1, \ldots, i_p \in \mathcal{A} \setminus \mathcal{A}_0$;
- $\mathsf{Loc}_0(i_j) = \mathsf{Loc}(v_j)$ for $j \in \{1, \ldots, p\}$.

Proposition 6. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_o, p_0)$ be a topology, $K_0 = (\mathcal{P}_0; \Phi_0; t_0)$ be a configuration built on $\mathcal{T}_0$, and V be a set of agents such that:*

$$\{a \mid \lfloor P \rfloor_a^t \in \mathcal{P}_0 \text{ or } \lfloor \Phi_0 \rfloor_a^t \neq \emptyset\} \subseteq V \subseteq \mathcal{A}_0.$$

Let $K = (\mathcal{P}; \Phi; t)$ be a configuration such that $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K$. We have that $(\mathcal{P}_0; \Phi_0; t_0) \xrightarrow{\mathsf{tr}}_{\mathcal{T}_{\mathsf{Loc}_V}} (\mathcal{P}; \Phi; t)$ where $\mathsf{Loc}_V = \mathsf{Loc}_{0|V}$.

Proof. We show this result by induction on the length of the derivation $K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K$. The base case, i.e. tr is the empty trace, is trivial. To conclude, it is sufficient to show that:

$$K_1 \xrightarrow{a,\alpha}_{\mathcal{T}} K_2 \text{ implies } K_1 \xrightarrow{a,\alpha}_{\mathcal{T}'} K_2 \text{ where } \mathcal{T}' = (\mathcal{A}', \mathcal{M}', \mathsf{Loc}', v_0, p_0) = \mathcal{T}_{\mathsf{Loc}_V}.$$

We consider each rule of the semantics one by one. Actually, the only rule that depends on the underlying topology is the rule IN. In such a case, we have that $\alpha = \mathtt{in}^\star(u)$ for some u. Moreover, we denote $K_1 = (\mathcal{P}_1; \Phi_1; t_1)$ and following the notations introduced in Section 3.3, we know that there exist $b \in \mathcal{A}_0$ (the agent responsible of the corresponding output) and $t_b \leq t_1 - \mathsf{Dist}_{\mathcal{T}}(b, a)$ (the time at which the output has been triggered) that satisfy the conditions of the rule. We distinguish two cases:

1. In case $b \in \mathcal{A}_0 \setminus \mathcal{M}_0$, then we know that there exists $\mathsf{w} \in \mathit{img}(\lfloor \Phi \rfloor_b^{t_b})$ such that $\mathsf{w}\Phi = u$. By definition of V, we have that $b \in V$, and therefore $b \in \mathcal{A}' \setminus \mathcal{M}'$. The same rule applies for the same reason.
2. In case $b \in \mathcal{M}_0$, then we know that there exists a recipe R such that $R\Phi_1{\downarrow} = u$, and for all $\mathsf{w} \in \mathit{vars}(R)$ there exists $c \in \mathcal{A}_0$ such that $\mathsf{w} \in \mathit{dom}(\lfloor \Phi_1 \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$. We show that the same rule applies using the same recipe R. However, the agent responsible of the output will be the agent i_a such that $\mathsf{Loc}'(i_a) = \mathsf{Loc}'(a) = \mathsf{Loc}_0(a)$ (note that $a \in V$), and this output will be performed at time t_1 (instead of t_b). We have that $R\Phi_1{\downarrow} = u$. Now, let $\mathsf{w} \in \mathit{vars}(R)$. Let $c \in \mathcal{A}_0$ such that $\mathsf{w} \in \mathit{dom}(\lfloor \Phi_1 \rfloor_c^{t_b - \mathsf{Dist}_{\mathcal{T}}(c,b)})$. We have that $c \in V$. It remains to show that $\mathsf{w} \in \mathit{dom}(\lfloor \Phi_1 \rfloor_c^{t_1 - \mathsf{Dist}_{\mathcal{T}'}(c, i_a)})$. For this, it is actually sufficient to establish that $t_1 - \mathsf{Dist}_{\mathcal{T}'}(c, i_a) \geq t_b - \mathsf{Dist}_{\mathcal{T}}(c, b)$. We know that:

$$\begin{aligned} & t_1 - \mathsf{Dist}_{\mathcal{T}}(b, a) \geq t_b \\ \Rightarrow \quad & t_1 - \mathsf{Dist}_{\mathcal{T}}(b, a) - \mathsf{Dist}_{\mathcal{T}}(c, b) \geq t_b - \mathsf{Dist}_{\mathcal{T}}(c, b) \\ \Rightarrow \quad & t_1 - (\mathsf{Dist}_{\mathcal{T}}(b, a) + \mathsf{Dist}_{\mathcal{T}}(c, b)) \geq t_b - \mathsf{Dist}_{\mathcal{T}}(c, b) \\ \Rightarrow \quad & t_1 - \mathsf{Dist}_{\mathcal{T}}(c, a) \geq t_b - \mathsf{Dist}_{\mathcal{T}}(c, b) \end{aligned}$$

   The last implication comes from the triangle inequality for the distance. Then, we obtain the expected result since $\mathsf{Loc}'(i_a) = \mathsf{Loc}'(a)$, and $\mathsf{Dist}_{\mathcal{T}}(c, a) = \mathsf{Dist}_{\mathcal{T}'}(c, a)$ since $a, c \in V$.

This concludes the proof. □

We are now able to establish our main result regarding mafia fraud.

Theorem 2. *Let $\mathcal{I}_0$ be a template, $\mathcal{P}_{\mathsf{prox}}$ be a protocol $\mathcal{I}_0$-executable, and $V_0(z_0, z_1)$ be a parametrised role containing the special event $\mathsf{end}(z_0, z_1)$. We have that $\mathcal{P}_{\mathsf{prox}}$ admits a mafia fraud attack w.r.t. t_0-proximity, if and only if, there is an attack against t_0-proximity in the topology $\mathcal{T}^{t_0}_{\mathsf{MF}}$.*

Proof. Consider a topology $\mathcal{T}_0 = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ such that $\mathcal{P}_{\mathsf{prox}}$ admits an attack. Let $K_0 = (\mathcal{P}_0; \Phi_0; t_{\mathsf{init}})$ be an initial configuration that is valid for $\mathcal{P}_{\mathsf{prox}}$, V_0 w.r.t. $\mathcal{T}_0$ and $\mathcal{I}_0$ such that K_0 admits an attack w.r.t. t_0-proximity in $\mathcal{T}_0$. We consider the execution witnessing this fact:

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}_0} (\lfloor \mathsf{end}(v_0, p_0).P \rfloor^{t_a}_{v_0} \uplus \mathcal{P}; \Phi_0 \cup \Phi; t) \text{ with } \mathsf{Dist}_{\mathcal{T}_0}(v_0, p_0) \geq t_0.$$

To establish this result, we combine the previous lemmas to show that there exists a corresponding trace of execution in $\mathcal{T}^{t_0}_{\mathsf{MF}}$. As already announced, the proof is performed in three steps, and the first step is done in two stages.

Step 1-a. Applying Lemma 2 with $\mathcal{H} = \mathcal{A}_0 \setminus (\mathcal{M}_0 \cup \{v_0, p_0\})$, we obtain a topology $\mathcal{T}_1 = (\mathcal{A}_1, \mathcal{M}_1, \mathsf{Loc}_1, v_0, p_0)$ where $\mathcal{A}_1 = \mathcal{A}_0$, $\mathcal{M}_1 = \mathcal{A}_0 \setminus \{v_0, p_0\}$, and $\mathsf{Loc}_1 = \mathsf{Loc}_0$. We have that:

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}_1} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_a}_{v_0} \uplus \mathcal{P}; \Phi_0 \cup \Phi; t)$$

with $\mathsf{Dist}_{\mathcal{T}_1}(v_0, p_0) = \mathsf{Dist}_{\mathcal{T}_0}(v_0, p_0) \geq t_0$.

We now consider the configuration $K_0^+ = (\mathcal{P}_0; \Phi_0^+; t_{\mathsf{init}})$ where we extend Φ_0 into Φ_0^+ such that $img(\lfloor \Phi_0^+ \rfloor^{t_{\mathsf{init}}}_a) = \mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_1)$ when $a \in \mathcal{M}_1$. Since we only increase the knowledge, we have that:

$$K_0^+ \xrightarrow{\mathsf{tr}}_{\mathcal{T}_1} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_a}_{v_0} \uplus \mathcal{P}; \Phi_0^+ \cup \Phi; t)$$

with $\mathsf{Dist}_{\mathcal{T}_1}(v_0, p_0) = \mathsf{Dist}_{\mathcal{T}_0}(v_0, p_0) \geq t_0$.

Step 1-b. Since the configuration K_0 is valid, we know that for all $\lfloor P \rfloor^{t_a}_{a_0} \in \mathcal{P}_0$, there exists a role $Q \in \mathcal{P}_{\mathsf{prox}}$ such that $P = Q\tau$ with $\tau = \{z_0 \mapsto a_0, \ldots, z_n \mapsto a_n\}$, $z_0, \ldots, z_n \in \mathcal{Z}$, and $a_0, \ldots, a_n \in \mathcal{A}_0 = \mathcal{A}_1$. Moreover, we know that for any term u occurring in an output or a let construction in P, there exists a corresponding term u' occurring in an output or a let construction in Q and such that $u = u'\tau$. Let $\mathcal{I}_0 = \{v_1, \ldots, v_k\}$, and $\sigma = \{\mathsf{w}_1 \mapsto v_1, \ldots, \mathsf{w}_k \mapsto v_k\}$. Since Q is $\mathcal{I}_0$-executable, there exists a term $R \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, \{\mathsf{w}_1, \ldots, \mathsf{w}_k\} \cup \mathcal{N} \cup \mathcal{X})$ such that $u' = R\sigma\downarrow$. Thus, we know that $u\downarrow = u'\tau\downarrow = R\sigma\downarrow\tau\downarrow = R\sigma\tau\downarrow$. Actually, we have that $img(\sigma\tau) \subseteq \mathsf{Knows}(\mathcal{I}_0, a_0, \mathcal{A}_1)$. Therefore, since $\mathsf{Knows}(\mathcal{I}_0, a_0, \mathcal{A}_1) \subseteq img(\lfloor \Phi_0^+ \rfloor^{t_{\mathsf{init}}}_{a_0})$, we have that there exists $R' \in \mathcal{T}(\Sigma^+_{\mathsf{pub}}, \mathcal{W} \cup \mathcal{N} \cup \mathcal{X})$ such that $u\downarrow = R' \lfloor \Phi_0^+ \rfloor^{t_{\mathsf{init}}}_{a_0}\downarrow$. Now, we can apply Lemma 3 with $K_0^+ = (\mathcal{P}_0; \Phi_0^+; t_{\mathsf{init}})$ and $\mathcal{D} = \mathcal{M}_1$ and conclude that:

- $(\tilde{\mathcal{P}}_0; \Phi_0^+; t_{\mathsf{init}}) \xrightarrow{\mathsf{tr}\rho_2}_{\mathcal{T}_1} (\lfloor \mathsf{end}(v_0, p_0) \rfloor^{t_a}_{v_0} \uplus \tilde{\mathcal{P}}\rho_2; \Phi_0^+ \cup \Phi\rho_2; t)$ where $\tilde{\mathcal{P}}$ is the mutiset of extended processed obtained from $\mathcal{P}$ by removing new instructions of the form `new` n and replacing the occurrence of n by c_0 in processes located in $a \in \mathcal{D} = \mathcal{M}_1$, and ρ_2 replaces any name that has been generated to trigger a rule NEW executed by an agent in $\mathcal{D}$ by c_0.

– any $\lfloor P \rfloor_{a_0}^{t_a} \in \tilde{\mathcal{P}}_0$ with $a_0 \in \mathcal{M}_1$ does not contain new construction and is executable w.r.t. $\lfloor \Phi_0^+ \rfloor_{a_0}^{t_{\mathsf{init}}}$ because $c_0 \in \Sigma_0 \subseteq \Sigma_{\mathsf{pub}}^+$ and therefore R' in which we replace any occcurrence of a name in $\mathcal{N}$ by c_0 is a recipe in $\mathcal{T}(\Sigma_{\mathsf{pub}}^+, \lfloor \Phi_0^+ \rfloor_{a_0}^{t_{\mathsf{init}}} \cup \mathcal{X})$ that allows one to conclude.

We are now able to apply Proposition 5 with $\mathcal{D}_0 = \mathcal{M}_1$ and we have that:

$$(\overline{\mathcal{P}_0}; \Phi_0^+; t_{\mathsf{init}}) \xrightarrow{\overline{\mathsf{tr}\rho_2}}_{\mathcal{T}_1} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_a} \uplus \overline{\mathcal{P}\rho_2}; \Phi_0^+ \cup \overline{\Phi\rho_2}; t) \text{ with } \mathsf{Dist}_{\mathcal{T}_1}(v_0, p_0) \geq t_0.$$

where $\overline{\mathcal{P}\rho_2}$ (resp. $\overline{\Phi\rho_2}$, $\overline{\mathsf{tr}\rho_2}$) is obtained from $\mathcal{P}\rho_2$ (resp. $\Phi\rho_2$, $\mathsf{tr}\rho_2$) by removing processes (frame elements, actions) located in $a \in \mathcal{D}_0 = \mathcal{M}_1$. We may note that the transformation $\tilde{\cdot}$ has some effect on processes that are removed by the transformation $\overline{\cdot}$, and thus can easily be removed.

Step 2. We now consider Φ_0^{++} the same as Φ_0^+ but frame elements located at $a \in \mathcal{M}_1$ are moved to v_0. Let $\delta_0 = Max(\mathsf{Dist}_{\mathcal{T}_1}(v_0, a))$ for any $a \in \mathcal{M}_1$. Clearly, we have that:

$$(\overline{\mathcal{P}_0}; \Phi_0^{++}; t_{\mathsf{init}} + \delta_0) \xrightarrow{\overline{\mathsf{tr}\rho_2}}_{\mathcal{T}_1} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_a} \uplus \overline{\mathcal{P}\rho_2}; \Phi_0^{++} \cup \mathsf{Shift}(\overline{\Phi\rho_2}, \delta_0); t + \delta_0)$$

with $\mathsf{Dist}_{\mathcal{T}_1}(v_0, p_0) \geq t_0$. Indeed, shifting by $+\delta_0$ the initial configuration, all the messages moved from an agent $a \in \mathcal{M}_1$ to v_0 can be used by a at the time t_{init} of the initial frame. Then we can apply Proposition 6 on $\mathcal{T}_1$ and $(\overline{\mathcal{P}_0}; \Phi_0^{++}; t_{\mathsf{init}} + \delta_0)$ with $V = \{v_0, p_0\}$. We deduce that (where $\mathcal{T}_1' = (\{v_0, p_0, i_P, i_V\}, \{i_P, i_V\}, \mathsf{Loc}_{\{v_0, p_0\}})$):

$$(\overline{\mathcal{P}_0}; \Phi_0^{++}; t_{\mathsf{init}} + \delta_0) \xrightarrow{\overline{\mathsf{tr}\rho_2}}_{\mathcal{T}_1'} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_a} \uplus \overline{\mathcal{P}\rho_2}; \Phi_0^{++} \cup \mathsf{Shift}(\overline{\Phi\rho_2}, \delta_0); t + \delta_0)$$

with $\mathsf{Dist}_{\mathcal{T}_1'}(v_0, p_0) = \mathsf{Dist}_{\mathcal{T}_1}(v_0, p_0) \geq t_0$.

Step 3. To reduce the size of the initial frame, now we apply Lemma 1 on $K_0' = (\overline{\mathcal{P}_0}; \Phi_0^{++}; t_{\mathsf{init}} + \delta_0)$ using $\rho_3 : \mathcal{A} \to \mathcal{A}_1' = \{v_0, p_0, i_P, i_V\}$ such that $\rho_3(a) = i_P$ for any $a \notin \mathcal{A}_1'$. We have that:

$$(\overline{\mathcal{P}_0}\rho_3; \Phi_0^{++}\rho_3; t_{\mathsf{init}} + \delta_0) \xrightarrow{\overline{\mathsf{tr}\rho_2}\rho_3}_{\mathcal{T}_1'} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_a} \uplus \overline{\mathcal{P}\rho_2}\rho_3; \Phi_0^{++}\rho_3 \cup \mathsf{Shift}(\overline{\Phi\rho_2}\rho_3, \delta_0); t + \delta_0)$$

with $\mathsf{Dist}_{\mathcal{T}_1'}(v_0, p_0) \geq t_0$.

Now, we show that $(\overline{\mathcal{P}_0}\rho_3; \Phi_0^{++}\rho_3; t_{\mathsf{init}} + \delta_0)$ is almost valid. Indeed to turn this configuration into a valid one, we simply have to move frame elements (those that we have added during Step 3) located in v_0 to i_V. This will not change the underlying execution since both nodes are located at the exact same place. Then, we add some frame elements in i_P, more precisely $\mathsf{Knows}(\mathcal{I}_0, i_P, \mathcal{A}_1')$. These additional elements will not alter the underlying execution.

We may note that $\overline{\mathcal{P}_0}\rho_3$ satisfies the first item of Definition 2: since $\mathcal{P}_0$ is valid, there exits t' such that $\lfloor V_0(v_0, p_0) \rfloor_{v_0}^{t'} \in \mathcal{P}_0$ and by construction $\overline{\mathcal{P}_0}\rho_3$ still contains $\lfloor V_0(v_0, p_0) \rfloor_{v_0}^{t'}$. Moreover, for all $\lfloor P' \rfloor_{a'}^{t'} \in \overline{\mathcal{P}_0}\rho_3$, by construction,

we know that $a' \in \{v_0, p_0\} = \mathcal{A}'_1 \setminus \mathcal{M}'_1$. Finally, we have that there exists $\lfloor P'' \rfloor_{a'}^{t'} \in \mathcal{P}_0$ such that $P'' = P(a', a_1, \ldots, a_n)$ for $P \in \mathcal{P}_{\mathsf{prox}}$ and $P' = P''\rho_3 = P(a', \rho_3(a_1), \ldots, \rho_3(a_n))$ with $\rho_3(a_1), \ldots, \rho_3(a_n) \in \mathcal{A}'_1$.

To conclude, we have to shorten the distance between v_0, i_V and p_0, i_P. To do so, we consider a new topology $\mathcal{T} = (\mathcal{A}'_1, \mathcal{M}'_1, \mathsf{Loc}, v_0, p_0)$ such that $\mathsf{Loc}(v_0) = \mathsf{Loc}(i_v)$, $\mathsf{Loc}(p_0) = \mathsf{Loc}(i_p)$ and $\mathsf{Dist}_{\mathcal{T}}(v_0, p_0) = t_0$. □

D Proof of Theorem 3 (Distance hijacking attack)

To establish this result, we will first show that we can transform any attack trace into an attack trace having a specific format. For this, we need to show that some actions can be swapped without breaking the underlying action, and thus preserving the fact that the trace is an attack.

We first introduce annotations in the semantics of our processes, in order to ease their analysis.

D.1 Annotations

We shall now define an annotated semantics whose transitions are equipped with more informative actions. The annotated actions will feature labels identifying which process in the multiset has performed the action (session identifier). This will allow us to identify which specific agent performed some action. We also put in the annotation the global time at which the action has been done. In case of an output, the annotation will indicate the name of the handle w that has been used to store the output in the frame. In case of an input, the annotation will indicate by a triple (b, t_b, R) the name b of the agent responsible of the corresponding output, the time at which this output has been performed, as well as the recipe R used to build this output.

Formally, an action is either empty (for the TIM rule) or of the form a, α with $\alpha \in \{\tau, \mathtt{out}(u), \mathtt{in}^\star(u)\}$, and thus an annotated action is:

- empty for the TIM rule;
- $(a, \alpha, s, t, \mathsf{w})$ when the underlying action a, α is of the form $a, \mathtt{out}(u)$. In such a case, s is the session identifier of the agent responsible of this action, t is the global time at which this output has been done, and w is the handle added in the frame.
- $(a, \alpha, s, t, (b, t_b, R))$ when the underlying action a, α is of the form $a, \mathtt{in}^\star(u)$. In such a case, s is the session identifier of the agent responsible of this action, t is the global time at which this input has been done, b is the agent responsible of the corresponding output, t_b the time at which this output has been done ($t_b \leq t$), and R the recipe that has been used to forge this output.
- $(a, \alpha, s, t, \emptyset)$ otherwise.

In the relaxed semantics, similar annotations can be added. Of course, in such a case, information about time is not relevant. Thus annotations are either of the form $(a, \alpha, s, \mathsf{w})$ (case of the output) or of the form $(a, \alpha, s, (b, R))$ (case of the input), or $(a, \alpha, s, \emptyset)$ otherwise. Note that in the relaxed semantics, the TIM rule does not exist.

Lemma 4. *Let $\mathcal{T}$ be a topology, and $K_0 \overset{L_1}{\rightsquigarrow}_{\mathcal{T}} \cdots \overset{L_n}{\rightsquigarrow}_{\mathcal{T}} K_n$ be an execution such that for all $L_i = (a_i, \alpha_i, s_i, r_i)$ we have that $\alpha_i \neq \mathtt{in}^{<t}(x)$ for some $\mathtt{t}$. Let K'_0 be a configuration such that $\mathsf{untimed}(K'_0) = K_0$. We have that there exists a configuration K'_n such that $\mathsf{untimed}(K'_n) = K_n$, and $K'_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} K'_n$ with $\mathsf{tr} = (a_1, \alpha_1) \dots (a_n, \alpha_n)$.*

Proof. Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ be a topology. This proof is quite straightforward and can be formally done by induction on the length of the execution. Before each application of the rule IN, we apply the TIM rule with a delay $\delta = \max_{a,b \in \mathcal{A}_0}(\mathsf{Dist}_{\mathcal{T}}(a, b))$ to be sure that all the messages necessary to build the inputted term are available. □

D.2 Permutations of independent actions

Definition 9. *Given an execution $K_0 \overset{L_1}{\rightsquigarrow}_{\mathcal{T}} \cdots \overset{L_n}{\rightsquigarrow}_{\mathcal{T}} K_n$ with $L_i = (a_i, \alpha_i, s_i, r_i)$, $L_j = (a_j, \alpha_j, s_j, r_j)$, we say that L_j is* dependent *of L_i, denoted $L_j \hookrightarrow L_i$, if $i < j$, and:*

- *either $s_i = s_j$ (and thus $a_i = a_j$), and in that case L_j is* sequentially-dependent *of L_i, denoted $L_j \hookrightarrow_s L_i$;*
- *or $\alpha_i = \mathtt{out}(v)$, $\alpha_j = \mathtt{in}^\star(u)$, and $r_i \in vars(r_j)$, and in that case L_j is* data-dependent *of L_i, denoted $L_j \hookrightarrow_d L_i$.*

We note $L_j \not\hookrightarrow L_i$ when L_j is not dependent of L_i, i.e. $L_j \not\hookrightarrow_s L_i$ and $L_j \not\hookrightarrow_d L_i$.

Lemma 5. *Given a topology $\mathcal{T}$, and an execution $K_0 \overset{L_1}{\rightsquigarrow}_{\mathcal{T}} K_1 \overset{L_2}{\rightsquigarrow}_{\mathcal{T}} K_2$ such that $L_2 \not\hookrightarrow L_1$. We have that $K_0 \overset{L_2}{\rightsquigarrow}_{\mathcal{T}} K_i \overset{L_1}{\rightsquigarrow}_{\mathcal{T}} K_2$ for some K_i.*

Proof. Let $K_0 = (\mathcal{P}_0; \Phi_0)$, $K_1 = (\mathcal{P}_1; \Phi_1)$, and $K_2 = (\mathcal{P}_2; \Phi_2)$ be three untimed configurations such that $K_0 \overset{L_1}{\rightsquigarrow}_{\mathcal{T}} K_1 \overset{L_2}{\rightsquigarrow}_{\mathcal{T}} K_2$ with $L_2 \not\hookrightarrow L_1$. Let $L_1 = (a_1, \alpha_1, s_1, r_1)$ and $L_2 = (a_2, \alpha_2, s_2, r_2)$. By hypothesis, we know that $L_2 \not\hookrightarrow L_1$, and thus we have that $s_1 \neq s_2$. Therefore, we have that:

- $\mathcal{P}_0 = \lfloor c_1.P_1 \rfloor_{a_1} \cup \lfloor c_2.P_2 \rfloor_{a_2} \uplus \mathcal{Q}$,
- $\mathcal{P}_1 = \lfloor P'_1 \rfloor_{a_1} \cup \lfloor c_2.P_2 \rfloor_{a_2} \uplus \mathcal{Q}$,
- $\mathcal{P}_2 = \lfloor P'_1 \rfloor_{a_1} \cup \lfloor P'_2 \rfloor_{a_2} \uplus \mathcal{Q}$

where $c_1, c_2 \in \{\mathtt{in}^\star(x),\ \mathtt{new}\ n,\ \mathtt{out}(u),\ \mathtt{reset},\ \mathtt{let}\ x = v\ \mathtt{in}\ \}$.

First, in case $\alpha_1 = \mathtt{out}(v)$ and $\alpha_2 = \mathtt{in}^\star(u)$, we have that $\Phi_2 = \Phi_1 = \Phi_0 \uplus \{r_1 \xrightarrow{a_1} v\}$ and since $L_2 \not\hookrightarrow_d L_1$, we know that $r_1 \notin vars(r_2)$. Therefore, we

deduce that $vars(r_2) \subseteq dom(\Phi_0)$. Now, let $K_i = (\lfloor c_1.P_1 \rfloor_{a_1} \uplus \lfloor P_2' \rfloor_{a_2} \uplus \mathcal{Q}; \Phi_0)$. Relying on the fact that $r_1 \notin vars(r_2)$ in case $\alpha_1 = \mathtt{out}(v)$ and $\alpha_2 = \mathtt{in}^\star(u)$, it is easy to see that $K_0 \overset{L_2}{\rightsquigarrow_{\mathcal{T}}} K_i \overset{L_1}{\rightsquigarrow_{\mathcal{T}}} K_2$. □

Corollary 1. *We consider a topology $\mathcal{T}$, an execution $K_0 \overset{L_1}{\rightsquigarrow_{\mathcal{T}}} \cdots \overset{L_n}{\rightsquigarrow_{\mathcal{T}}} K_n$, and two sets I and J such that $I = \{i_1, \ldots, i_p\}$, $J = \{j_1, \ldots, j_q\}$, $I \uplus J = \{0, \ldots, n\}$, $i_1 < \cdots < i_p$, and $j_1 < \cdots < j_q$. Moreover, we assume that $L_i \not\rightarrow L_j$ for any $i \in I$, and $j \in J$. In such a case, we have that:*

$$K_0 \overset{L_{i_1}}{\rightsquigarrow_{\mathcal{T}}} \cdots \overset{L_{i_p}}{\rightsquigarrow_{\mathcal{T}}} \overset{L_{j_1}}{\rightsquigarrow_{\mathcal{T}}} \cdots \overset{L_{j_q}}{\rightsquigarrow_{\mathcal{T}}} K_n$$

Proof. Given a trace $K_0 \overset{L_1}{\rightsquigarrow} \cdots \overset{L_n}{\rightsquigarrow} K_n$ and the two ordered sets I and J, we can translate the trace as a two-letters (i and j) word, each transition either converted into i or j depending on the set in which its index belongs. We consider the lexicographical order induced by the fact that $i < j$, *e.g.* $iijjii < ijijii$.

Let us suppose that there exists a trace satisfying all the hypotheses but such that the result does not hold. We consider the minimal one in the sense of the lexicographical order of the corresponding 2-letters word. This trace is a word w of the form $\cdots ji \cdots$, and the portion of it corresponding to the pattern ji is:

$$K_0' \overset{L_{j_{q'}}}{\rightsquigarrow} K_1' \overset{L_{i_{p'}}}{\rightsquigarrow} K_2'$$

for some q' and some p'. Since $L_{i_{p'}} \not\rightarrow L_{j_{q'}}$, we can use Lemma 5, and rewrite this trace as $K_0' \overset{L_{i_{p'}}}{\rightsquigarrow} K_i' \overset{L_{j_{q'}}}{\rightsquigarrow} K_2'$. Inserting such a trace into the whole exection trace we had at the beginning, we obtain a trace whose corresponding word w' is smaller than w (according to the lexicographical order), and this would lead to a contradiction. □

D.3 Proof of Proposition 2

Given a configuration K, $\phi(K)$ denotes its second component.

Proposition 7. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ be a topology, and consider an execution trace of the following form:*

$$K_0 \xrightarrow{L_{\mathtt{reset}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_1}_{\mathcal{T}} K_{\mathtt{out}} \xrightarrow{L_{\mathtt{out}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_2}_{\mathcal{T}} K_{\mathtt{in}} \xrightarrow{L_{\mathtt{in}}}_{\mathcal{T}} K_1$$

where $L_{\mathtt{reset}} = (v_0, \mathtt{reset}, s_0, t_{\mathtt{reset}}, \emptyset)$, $L_{\mathtt{out}} = (v_0, \mathtt{out}(v), s_0, t_{\mathtt{out}}, \mathsf{w}_0)$, and $L_{\mathtt{in}} = (v_0, \mathtt{in}^{<t_g}(u), s_0, t_{\mathtt{in}}, (b_0, t_b^0, R_0))$, and these are the only actions executed with the session identifier s_0.

Then, there exists an execution trace:

$$K_0' \xrightarrow{\mathsf{tr}_0'}_{\mathcal{T}} K_{\mathtt{reset}}' \xrightarrow{L_{\mathtt{reset}}'}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_1'}_{\mathcal{T}} K_{\mathtt{out}}' \xrightarrow{L_{\mathtt{out}}'}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_2'}_{\mathcal{T}} K_{\mathtt{in}}' \xrightarrow{L_{\mathtt{in}}'}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_3'}_{\mathcal{T}} K_1'$$

where:

– K'_0 *(resp.* K'_1*) is the untimed counterpart of* K_0 *(resp.* K_1*), i.e.* $K'_i = \mathsf{untimed}(K_i)$ *with* $i \in \{0,1\}$*;*
– $L'_{\mathtt{reset}}$*,* $L'_{\mathtt{out}}$*, and* $L'_{\mathtt{in}}$ *are the untimed counterpart of the annotations* $L_{\mathtt{reset}}$*,* $L_{\mathtt{out}}$*, and* $L_{\mathtt{in}}$*, i.e.* $L'_{\mathtt{reset}} = (v_0, \mathtt{reset}, s_0, \emptyset)$*,* $L'_{\mathtt{out}} = (v_0, \mathtt{out}(v), s_0, \mathsf{w}_0)$*, and* $L'_{\mathtt{in}} = (v_0, \mathtt{in}^{<t_g}(u), s_0, (b_0, R_0))$*;*
– *for any* $L = (a, \alpha, s, r)$ *occurring in* $\mathsf{tr}'_1.L'_{\mathtt{out}}.\mathsf{tr}'_2.L'_{\mathtt{in}}$*,* $2\mathit{Dist}_{\mathcal{T}}(v_0, a) < t_g$*. Moreover, when* $\alpha = \mathtt{in}^\star(u)$*, we have that* $r = (b', R')$ *for some* b' *and some* R' *such that* $2\mathit{Dist}_{\mathcal{T}}(v_0, b') < t_g$ *or* $\mathit{vars}(R') \subseteq \mathit{dom}(\phi(K'_{\mathtt{reset}}))$*.*

Proof. Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ be a topology and

$$K_0 \xrightarrow{L_{\mathtt{reset}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_1}_{\mathcal{T}} K_{\mathtt{out}} \xrightarrow{L_{\mathtt{out}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_{2^<}}_{\mathcal{T}} K \xrightarrow{\mathsf{tr}_{2^\geq}}_{\mathcal{T}} K_{\mathtt{in}} \xrightarrow{L_{\mathtt{in}}}_{\mathcal{T}} K_1$$

be a trace as defined in the proposition such that $\mathsf{tr}_2 = \mathsf{tr}_{2^<}\,\mathsf{tr}_{2^\geq}$; $t < t_{\mathtt{out}} + t_g/2$ for any $L = (a, \alpha, s, t, r)$ occurring in $\mathsf{tr}_{2^<}$, and $t \geq t_{\mathtt{out}} + t_g/2$ for any $L = (a, \alpha, s, t, r)$ occurring in $\mathsf{tr}_{2^\geq}$.

By Definition of the relaxed semantics, we have that

$$K'_0 \overset{L'_{\mathtt{reset}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\overline{\mathsf{tr}_1}}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\overline{\mathsf{tr}_{2^<}}}{\rightsquigarrow}_{\mathcal{T}} K' \overset{\overline{\mathsf{tr}_{2^\geq}}}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow}_{\mathcal{T}} K'_1$$

where K'_0 (resp. K'_1) is the untimed counterpart of K_0 (resp. K_1), $L'_{\mathtt{reset}}$, $L'_{\mathtt{out}}$, and $L'_{\mathtt{in}}$ are the untimed counterpart of the annotations $L_{\mathtt{reset}}$, $L_{\mathtt{out}}$, and $L_{\mathtt{in}}$ and $\overline{\mathsf{tr}_1}$ (resp. $\overline{\mathsf{tr}_{2^<}}$, $\overline{\mathsf{tr}_{2^\geq}}$) is the untimed counterpart of tr_1 (resp. $\mathsf{tr}_{2^<}$, $\mathsf{tr}_{2^\geq}$). In the following, we consider:

– $\mathsf{tr} = L_{\mathtt{reset}}.\mathsf{tr}_1.L_{\mathtt{out}}.\mathsf{tr}_{2^<}.\mathsf{tr}_{2^\geq}.L_{\mathtt{in}}$ the timed trace, and
– $\overline{\mathsf{tr}} = L'_{\mathtt{reset}}.\overline{\mathsf{tr}_1}.L'_{\mathtt{out}}.\overline{\mathsf{tr}_{2^<}}.\overline{\mathsf{tr}_{2^\geq}}.L'_{\mathtt{in}}$ its untimed counterpart.

Finally, we define $\mathsf{Close}(v_0) = \{a \in \mathcal{A}_0 \mid 2\mathsf{Dist}_{\mathcal{T}}(v_0, a) < t_g\}$.

In order to obtain an execution trace in the relaxed semantics that satisfies all the requirements, we proceed in two main steps. We first show that we can push the actions from $\mathsf{tr}_{2^<}$ that do not satisfy the requirements before $L_{\mathtt{reset}}$, and then we will establish that we can push those from $\mathsf{tr}_{2^\geq}$ after $L_{\mathtt{in}}$.

Step 1: cleaning of $\mathsf{tr}_{2^<}$. Let S^1_a be the set of actions in $\overline{\mathsf{tr}}$ such that their corresponding timed action in tr is executed by agents $a \notin \mathsf{Close}(v_0)$ and before $t_{\mathtt{out}} + t_g/2$. Formally, we have that:

$$S^1_a = \{(a, \alpha, s, r) \in \overline{\mathsf{tr}} \mid a \notin \mathsf{Close}(v_0) \text{ and } (a, \alpha, s, t, r') \in \mathsf{tr} \text{ and } t < t_{\mathtt{out}} + t_g/2\}.$$

We also define S^2_a the set of actions in $\overline{\mathsf{tr}}$ such that their corresponding timed action in tr is an input for which the inputted message has been forged by an agent $b \notin \mathsf{Close}(v_0)$ at a time $t_b < t_{\mathtt{out}} + t_g/2$. Formally, we have that:

$$S^2_a = \left\{ (a, \mathtt{in}^\star(u), s, r) \in \overline{\mathsf{tr}} \;\middle|\; \begin{array}{l} L = (a, \mathtt{in}^\star(u), s, t, (b, t_b, R)) \in \mathsf{tr} \\ b \notin \mathsf{Close}(v_0),\ t_b < t_{\mathtt{out}} + t_g/2. \end{array} \right\}$$

Then, we build $\mathsf{BD}(S^1_a, S^2_a)$ as follows:

1. $S := S_a^1$.
2. For each $\gamma \in S_a^2$, for all transition β in $\overline{\mathsf{tr}}$, if $\gamma \hookrightarrow_d \beta$, then $S := S \cup \{\beta\}$.
3. For each $\gamma \in S$, for all transition β in $\overline{\mathsf{tr}}$ if $\gamma \hookrightarrow \beta$, then $S := S \cup \{\beta\}$.
4. $\mathsf{BD}(S_a^1, S_a^2) := S$.

We may note that at step 3, a fix point will be reached since we are working on a finite trace $\overline{\mathsf{tr}}$. We may also note that $\mathsf{BD}(S_a^1, S_a^2)$ only contains actions from $\mathsf{tr}_{2<}$. Indeed, let $L_1 = (a_1, \alpha_1, s_1, t_1, (b_1, t_b^1, R_1))$ be such that its untimed counterpart L_1' belongs to S_a^2. Let us consider $L_2 = (a_2, \alpha_2, s_2, t_2, \mathsf{w}_2)$ such that its untimed counterpart L_2' satisfies $L_1' \hookrightarrow_d L_2'$ then we can deduce that

$$t_2 \leq t_b^1 - \mathsf{Dist}_{\mathcal{T}}(a_2, b_1) \leq t_b^1 < t_{\mathtt{out}} + t_g/2.$$

Therefore, we can deduce that for all action $L' \in \mathsf{BD}(S_a^1, S_a^2)$, and this allows us to conclude.

Let $\overline{\mathsf{BD}(S_a^1, S_a^2)}$ be the set of transitions β such that $\beta \notin \mathsf{BD}(S_a)$ and $\beta \in L'_{\mathtt{reset}}.\overline{\mathsf{tr}_1}.L'_{\mathtt{out}}.\overline{\mathsf{tr}_{2<}}$.

Claim. 1. For all $\alpha \in \mathsf{BD}(S_a^1, S_a^2)$ and $\beta \in \overline{\mathsf{BD}(S_a^1, S_a^2)}$, we have that $\alpha \not\hookrightarrow \beta$.

Proof. By construction of $\mathsf{BD}(S_a^1, S_a^2)$.

Claim. 2. We have that $L'_{\mathtt{reset}} \in \overline{\mathsf{BD}(S_a^1, S_a^2)}$.

Proof. If $L'_{\mathtt{reset}} \in \mathsf{BD}(S_a^1, S_a^2)$, then we must have that $L'_{\mathtt{out}} \in \mathsf{BD}(S_a^1, S_a^2)$. Therefore, there exists a sequence of actions such that $L_1' \hookrightarrow \cdots \hookrightarrow L_n' \hookrightarrow L'_{\mathtt{out}}$ with $L_1' \in S_a^1 \cup S_a^2$. For $i \in \{1, \ldots, n\}$, we note a_i the name of the agent executing L_i' and t_i the time at which the timed counterpart of this action occurs in tr. We distinguish two cases:

- $L_1' \in S_a^1$. We have that:

$$t_1 - t_{\mathtt{out}} \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_n) + \sum_{i=1}^{n-1} \mathsf{Dist}_{\mathcal{T}}(a_i, a_{i+1}) \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_1) \geq t_g/2$$

  since we know that $a_1 \notin \mathsf{Close}(v_0)$ by Definition of S_a^1. We also have that $t_1 < t_{\mathtt{out}} + t_g/2$ by Definition of S_a^1, and thus $t_1 - t_{\mathtt{out}} < tg/2$, yielding to a contradiction.
- $L_1' = (a_1, \mathtt{in}^\star(u), s_1, (b_1, R_1)) \in S_a^2$. Let $L_1 = (a_1, \mathtt{in}^\star(u), s_1, t_1, (b_1, t_b^1, R_1))$ its counterpart action in tr. By construction, we have that $L_1' \hookrightarrow_d L_2'$. According to the semantics of the rule IN, we have that $t_b^1 \geq t_2 + \mathsf{Dist}_{\mathcal{T}}(a_2, b_1)$. We also know that $t_2 - t_{\mathtt{out}} \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_2)$. Thus we have $t_b^1 - t_{\mathtt{out}} \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_2) + \mathsf{Dist}_{\mathcal{T}}(a_2, b_1) \geq \mathsf{Dist}_{\mathcal{T}}(v_0, b_1)$. By Definition of S_a^2, we know that $b_1 \notin \mathsf{Close}(v_0)$ and thus $\mathsf{Dist}_{\mathcal{T}}(v_0, b_1) \geq t_g/2$, and thus $t_b^1 - t_{\mathtt{out}} \geq t_g/2$. However, by definition of S_a^2, we know that $t_b^1 < t_{\mathtt{out}} + t_g/2$, yielding to a contradiction.

This allows us to conclude the proof of this claim. □

With Claim 1 and Claim 2, we can apply Corollary 1, and thus, we can move all transitions in $\mathsf{BD}(S_a^1, S_a^2)$ right before $L'_{\mathtt{reset}}$, leading us to a partially-cleaned trace $\widetilde{\mathsf{tr}}$:

$$K'_0 \overset{\widetilde{\mathsf{tr}_0}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_1}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_{2\leq}}}{\rightsquigarrow_{\mathcal{T}}} K' \overset{\overline{\mathsf{tr}_{2\geq}}}{\rightsquigarrow_{\mathcal{T}}} K'_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow_{\mathcal{T}}} K'_1$$

Step 2: cleaning of $\mathsf{tr}_{2\geq}$. We proceed in a similar way to clean the other part of the trace. Let S_b^1 be the set of actions in the relaxed trace $\overline{\mathsf{tr}}$ such that their corresponding action in the initial trace is executed by agents who are not in $\mathsf{Close}(v_0)$ between $L_{\mathtt{reset}}$ and $L_{\mathtt{in}}$ and after $t_{\mathtt{out}} + t_g/2$. Formally, we have that:

$$S_b^1 = \{(a, \alpha, s, r) \in \overline{\mathsf{tr}} \mid a \notin \mathsf{Close}(v_0) \text{ and } (a, \alpha, s, t, r') \in \mathsf{tr} \text{ and } t \geq t_{\mathtt{out}} + t_g/2\}.$$

We also define S_b^2 the set of actions in $\overline{\mathsf{tr}}$ such that their corresponding timed action in tr is an input in which the inputted message has been forged by an agent $b \notin \mathsf{Close}(v_0)$ at a time $t_b \geq t_{\mathtt{out}} + t_g/2$. Formally, we have that:

$$S_b^2 = \left\{ (a, \mathtt{in}^\star(u), s, r) \in \overline{\mathsf{tr}} \;\middle|\; \begin{array}{l} L = (a, \mathtt{in}^\star(u), s, t, (b, t_b, R)) \in \mathsf{tr} \\ b \notin \mathsf{Close}(v_0),\ t_b \geq t_{\mathtt{out}} + t_g/2. \end{array} \right\}$$

Let $S_b = S_b^1 \cup S_b^2$, and we consider the set $\mathsf{FD}(S_b)$ built as follows:

1. $S := S_b$.
2. For each $\gamma \in S$, for all transition β in $\overline{\mathsf{tr}}$, if $\beta \hookrightarrow \gamma$, then $S := S \cup \{\beta\}$.
3. $\mathsf{FD}(S_b) := S$.

Let $\overline{\mathsf{FD}(S_b)}$ be the set of transitions β such that $\beta \notin \mathsf{FD}(S_b)$ and $\beta \in \overline{\mathsf{tr}_{2\geq}}.L_{\mathtt{in}}$. We can note that for all action in $L \in \mathsf{FD}(S_b)$, its counterpart timed action L is executed at time t such that $t \geq t_{\mathtt{out}} + t_g/2$.

Claim. 3. For all $\alpha \in \mathsf{FD}(S_b)$ and $\beta \in \overline{\mathsf{FD}(S_b)}$, we have that $\beta \not\hookrightarrow \alpha$.

Proof. By construction of $\mathsf{FD}(S_b)$.

Claim. 4. We have that $L_{\mathtt{in}} \in \overline{\mathsf{FD}(S_b)}$.

Proof. If $L'_{\mathtt{in}} \in \mathsf{FD}(S_b)$, then there exists a sequence of actions $L'_{\mathtt{in}} \hookrightarrow L'_1 \hookrightarrow \cdots \hookrightarrow L'_n$ with $L'_n \in S_b$. For $i \in \{1, \ldots, n\}$, we not a_i the name of the agent executing L'_i and t_i the time at which this action occurs in tr. We distinguish two cases:

– $L'_n \in S_b^1$. In such a case, we have that

$$t_{\mathtt{in}} - t_n \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_1) + \sum_{i=1}^{n-1} \mathsf{Dist}_{\mathcal{T}}(a_i, a_{i+1}) \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_n) \geq t_g/2$$

since we know that $a_n \notin \mathsf{Close}(v_0)$. By definition of S_b^1, we know that $t_n \geq t_{\mathtt{out}} + t_g/2$. Thus, we have that $t_{\mathtt{in}} \geq t_{\mathtt{out}} + t_g$. However, the semantics gives us $t_{\mathtt{in}} - t_{\mathtt{out}} < t_g$ yielding a contradiction.

– $L'_n = (a_n, \mathtt{in}^\star(u), s_n, (b_n, R_n)) \in S^2_b$. Let $L_n = (a_n, \mathtt{in}^\star(u), s_n, t_n, (b_n, t^n_b, R_n))$ its counterpart action in tr. We have that

$$t_{\mathtt{in}} - t_n \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_1) + \sum_{i=1}^{n-1} \mathsf{Dist}_{\mathcal{T}}(a_i, a_{i+1}) \geq \mathsf{Dist}_{\mathcal{T}}(v_0, a_n).$$

We also have that $t_n \geq t^n_b + \mathsf{Dist}_{\mathcal{T}}(b_n, a_n) \geq t_{\mathtt{out}} + t_g/2 + \mathsf{Dist}_{\mathcal{T}}(b_n, a_n)$. Therefore, we have that:

$$t_{\mathtt{in}} \geq t_{\mathtt{out}} + t_g/2 + \mathsf{Dist}_{\mathcal{T}}(b_n, a_n) + \mathsf{Dist}_{\mathcal{T}}(v_0, a_n) \geq t_{\mathtt{out}} + t_g/2 + \mathsf{Dist}_{\mathcal{T}}(v_0, b_n).$$

Since $b_n \notin \mathsf{Close}(v_0)$, we know that $\mathsf{Dist}_{\mathcal{T}}(v_0, b_n) \geq t_g/2$. Thus, we have that $t_{\mathtt{in}} \geq t_{\mathtt{out}} + t_g$. This yields to a contradiction since the semantics gives us $t_{\mathtt{in}} - t_{\mathtt{out}} < t_g$.

This allows us to conclude the proof of this claim. □

With Claim 3 and Claim 4, we can apply Corollary 1 on the trace $\overline{\mathsf{tr}}$, and thus, we can move all transitions in $\mathsf{FD}(S_b)$ right after $L_{\mathtt{in}}$ leading us to a partially cleaned trace:

$$K'_0 \overset{L'_{\mathtt{reset}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\overline{\mathsf{tr}_1}}{\rightsquigarrow_{\mathcal{T}}} K'_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\overline{\mathsf{tr}_{2<}}}{\rightsquigarrow_{\mathcal{T}}} K' \overset{\widetilde{\mathsf{tr}_{2\geq}}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_3}}{\rightsquigarrow} K'_1$$

Therefore, we have that:

$$K'_0 \overset{\widetilde{tr_0}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_1}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_{2<}}}{\rightsquigarrow_{\mathcal{T}}} K' \overset{\widetilde{\mathsf{tr}_{2\geq}}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_3}}{\rightsquigarrow_{\mathcal{T}}} K'_1.$$

To conclude, it remains to check that for any $L' = (a, \mathtt{in}^\star(u), s, r)$ occurring in $\widetilde{\mathsf{tr}_1}.L'_{\mathtt{out}}.\widetilde{\mathsf{tr}_{2<}}.\widetilde{\mathsf{tr}_{2\geq}}.L'_{\mathtt{in}}$, we have that $r = (b', R')$ for some b' and some R' such that $2\mathsf{Dist}_{\mathcal{T}}(v_0, b') < t_g$ or $vars(R') \subseteq dom(\phi(\widetilde{K'}_{\mathtt{reset}}))$.

Let us consider $L = (a, \mathtt{in}^\star(u), s, t, (b', t_b, R'))$ the timed counterpart of L, and assume that $2\mathsf{Dist}_{\mathcal{T}}(v_0, b') \geq t_g$. We distinguish two cases:

– *Case* $t_b < t_{\mathtt{out}} + t_g/2$. In such a case, we know that $L' \in S^2_a$. For all $\mathsf{w} \in vars(R')$, there exists $L'_{\mathsf{w}} \in \overline{tr}$ such that $L' \hookrightarrow_d L'_{\mathsf{w}}$. Therefore, by construction, $L'_{\mathsf{w}} \in \mathsf{BD}(S^1_a, S^2_a)$, and this action has been moved before $L'_{\mathtt{reset}}$. Thus, we have that $vars(R') \subseteq dom(\phi(K'_{\mathtt{reset}}))$.
– *Case* $t_b \geq t_{\mathtt{out}} + t_g/2$. In such a case, we know that $L' \in S^2_b \subseteq \mathsf{FD}(S_b)$ and thus has been moved after $L'_{\mathtt{in}}$.

This concludes the proof. □

Proposition 2. *Let K_0 be a valid initial configuration for $\overline{\mathcal{P}_{\mathsf{prox}}}$ and V_0 w.r.t. a topology $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ and $\mathcal{I}_0$. If $K_0 \overset{\mathsf{tr}}{\rightsquigarrow_{\mathcal{T}}} K_1$ then there exists an execution $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow} K'_1$ such that $K'_i = \mathsf{untimed}(K_i)$ for $i \in \{0, 1\}$.*

Moreover, for any sub-execution of $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow} K'_1$ of the form

$$(\lfloor \mathtt{reset}.P \rfloor_{v_0} \uplus \mathcal{P}; \Phi_{\mathtt{reset}}) \overset{v_0,\tau}{\rightsquigarrow} (\lfloor P \rfloor_{v_0} \uplus \mathcal{P}; \Phi_{\mathtt{reset}}) \overset{\mathsf{tr}'_0}{\rightsquigarrow} K^-_{\mathtt{in}} \overset{v_0,\mathtt{in}^{<t}(u)}{\rightsquigarrow} K'_{\mathtt{in}}$$

where tr'_0 only contains actions (a, α) with $\alpha \in \{\tau, \mathtt{out}(u), \mathtt{in}(u)\}$, we have that:

– $2\mathit{Dist}_{\mathcal{T}}(v_0, a) < t$ *for any* $(a, \alpha) \in \mathsf{tr}'_0$*;*
– *for any* $(a, \mathtt{in}^\star(v))$ *occurring in* $\mathsf{tr}'_0.(v_0, \mathtt{in}^{<t}(u))$*, the agent* b *responsible of the output and the recipe* R *(as defined in Figure 1) are such that either* $2\mathit{Dist}_{\mathcal{T}}(v_0, b) < t$*, or* $vars(R) \subseteq dom(\Phi_{\mathtt{reset}})$*.*

Proof. Let K_0 be a valid initial configuration for $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. a topology $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ and $\mathcal{I}_0$. Let $K_0 \xrightarrow{\mathsf{tr}} K_1$ be a trace. Because of the specific grammar generating V_0 we can decompose tr into several pieces of the form: $L_{\mathtt{reset}}.\mathsf{tr}_1.L_{\mathtt{out}}.\mathsf{tr}_2.L_{\mathtt{in}}$ where $L_{\mathtt{reset}}$ is a reset transition, $L_{\mathtt{out}}$ the following output and $L_{\mathtt{in}}$ the following guarded input. Therefore, we can apply Proposition 7 on each piece. Finally we can concatenate all these relaxed sub-traces to obtain a full trace as described in Proposition 2. □

Theorem 3. *Let* $\mathcal{I}_0$ *be a template,* $\mathcal{P}_{\mathsf{prox}}$ *be a protocol,* $t_0 \in \mathbb{R}_+$*, and* $V_0(z_0, z_1)$ *be a parametrised role obtained using the following grammar:*

$$\begin{array}{rl} P, Q := \mathsf{end}(z_0, z_1) & \mid \mathtt{in}(x).P \quad \mid \mathtt{let}\ x = v\ \mathtt{in}\ P \\ \mid\ \mathtt{new}\ n.P & \mid \mathtt{out}(u).P \quad \mid \mathtt{reset}.\mathtt{out}(u').\mathtt{in}^{<t}(x).P \end{array}$$

where $x \in \mathcal{X}$*,* $n \in \mathcal{N}$*,* $u, u' \in \mathcal{T}(\Sigma_c^+, \mathcal{X} \cup \mathcal{N} \cup \{z_0, z_1\})$*,* $v \in \mathcal{T}(\Sigma^+, \mathcal{X} \cup \mathcal{N} \cup \{z_0, z_1\})$ *and* $t \leq 2 \times t_0$*. If* $\mathcal{P}_{\mathsf{prox}}$ *admits a distance hijacking attack w.r.t.* t_0*-proximity, then* $\overline{\mathcal{P}_{\mathsf{prox}}}$ *admits an attack against* t_0*-proximity in the topology* $\mathcal{T}_{\mathsf{DH}}^{t_0}$*.*

Proof. Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0) \in \mathcal{C}_{\mathsf{DH}}$ and $K_0 = (\mathcal{P}_0; \Phi_0; t_{\mathsf{init}})$ be a valid initial configuration for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$. We have that:

$$K_0 \xrightarrow{tr}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_v} \cup \mathcal{P}; \Phi; t) = K_1 \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Step 1: We first replace guarded inputs occurring in processes other than V_0 by simple inputs. Denoting $\overline{K_0}$ (resp. $\overline{K_1}$, $\overline{\mathsf{tr}}$) the counterpart of K_0 (resp. K_1 and tr) in which guarded inputs have been replaced by simple inputs, we have that:

$$\overline{K_0} \xrightarrow{\overline{tr}}_{\mathcal{T}} (\lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_v} \cup \overline{\mathcal{P}}; \Phi; t) = \overline{K_1} \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Indeed, all the required conditions to trigger a simple input will be satsified since a guarded input is like a simple input with a constraint regarding time. Moreoever, we have that $\overline{K_0}$ is a valid initial configuration for $\overline{\mathcal{P}_{\mathsf{prox}}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$.

Step 2: Let $\mathsf{Close}(v_0) = \{a \in \mathcal{A}_0 \mid \mathsf{Dist}_{\mathcal{T}}(v_0, a) < t_0\}$. Applying Proposition 2, we obtain a relaxed trace $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow}_{\mathcal{T}} K'_1$ such that $\mathsf{untimed}(\overline{K_i}) = K'_i$ for $i \in \{0, 1\}$. Moreover, for any sub-execution $K'_{\mathtt{reset}} \overset{v_0,\tau}{\rightsquigarrow} \overset{\mathsf{tr}'_0}{\rightsquigarrow} \overset{v_0,\mathtt{in}^{<t}(u)}{\rightsquigarrow} K'_{\mathtt{in}}$ of $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow} K'_1$ starting with an application of the rule RESET and whose only guarded input is the one occurring at the end, we have that:

– $2\mathsf{Dist}_{\mathcal{T}}(v_0, a) < t$ for any $(a, \alpha) \in \mathsf{tr}'_0$;

– for any $(a, \mathtt{in}^\star(v))$ occurring in $\mathsf{tr}'_0.(v_0, \mathtt{in}^{<t}(u))$, the agent b responsible of the output (as defined in Figure 4) is such that either $2\mathsf{Dist}_{\mathcal{T}}(v_0, b) < t$, or $vars(R) \subseteq dom(\phi(K'_{\mathtt{reset}}))$.

Since $t \leq 2 \times t_0$, we have that:

– $\mathsf{Dist}_{\mathcal{T}}(v_0, a) < t_0$ for any $(a, \alpha) \in \mathsf{tr}'_0$;
– for any $(a, \mathtt{in}^\star(v))$ occurring in $\mathsf{tr}'_0.(v_0, \mathtt{in}^{<t}(u))$, the agent b responsible of the output (as defined in Figure 4) is such that either $\mathsf{Dist}_{\mathcal{T}}(v_0, b) < t_0$, or $vars(R) \subseteq dom(\phi(K'_{\mathtt{reset}}))$.

Step 3: We consider the topology $\mathcal{T}' = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}', v_0, p_0)$ such that $\mathsf{Loc}'(v_0) = \mathsf{Loc}(v_0)$, $\mathsf{Loc}'(p_0)$ is such that $\mathsf{Dist}_{\mathcal{T}'}(v_0, p_0) = t_0$ and:

$$\mathsf{Loc}'(a) = \begin{cases} \mathsf{Loc}'(v_0) & \text{if } a \in \mathsf{Close}(v_0) \\ \mathsf{Loc}'(p_0) & \text{otherwise.} \end{cases}$$

In this topology, the agents far away from v_0 are moved to p_0, and agents in the neighborhood of v_0 are moved to v_0. In this topology $\mathcal{T}'$, we only have two locations, and we have that $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow}_{\mathcal{T}'} K'_1$ since the locations of the agents are no longer relevant in the relaxed semantics.

Step 4: We show that we can come back to a timed execution trace, i.e. one executable in the timed semantics by induction on the number of guarded inputs in the trace. Given a configuration $\widehat{K_0}$ such that $\mathsf{untimed}(\widehat{K_0}) = K'_0$, we show that there exists a configuration $\widehat{K_1}$ such that $\mathsf{untimed}(\widehat{K_1}) = K'_1$. To show this result, we split our execution trace $K'_0 \overset{\mathsf{tr}'}{\rightsquigarrow}_{\mathcal{T}'} K'_1$ on several blocks of actions: a block is either a trace with no guarded input, or a sequence of actions starting with a `reset` and ending at the first occurrence of a guarded input. To lift such a block in the timed semantics, we either rely on Lemma 4, or on the properties established at Step 2.

To conclude this step, it only remains to show how to lift a block in the timed semantics. Let $K'_{\mathtt{reset}} \overset{v_0,\tau}{\rightsquigarrow} \overset{\mathsf{tr}'_0}{\rightsquigarrow} \overset{v_0,\mathtt{in}^{<t}(u)}{\rightsquigarrow} K'_{\mathtt{in}}$ be such a block, and let $\widehat{K}$ be such that $\mathsf{untimed}(\widehat{K}) = K'_{\mathtt{reset}}$, and we denote $\widehat{t}$ the global time of configuration $\widehat{K}$. We have to show that there exists $\widehat{K_{\mathtt{in}}}$ such that $\widehat{K} \xrightarrow{(v_0,\tau).\mathsf{tr}'_0.(v_0,\mathtt{in}^{<t}(u))} \widehat{K_{\mathtt{in}}}$, and $\mathsf{untimed}(\widehat{K_{\mathtt{in}}}) = K'_{\mathtt{in}}$. We start by applying the rule TIM with the delay δ equals to $2 \times t_0$. Let $\widehat{K_+}$ be the resulting configuration.

Then we have to show that the sequence of actions $L_{\mathtt{reset}}.\mathsf{tr}'_1.L_{\mathtt{out}}.\mathsf{tr}'_2.L_{\mathtt{in}}$ can be executed without introducing any delay. Moreover, we show that the resulting configuration $\widehat{K_{\mathtt{in}}}$ is such that $\mathsf{untimed}(\widehat{K_{\mathtt{in}}}) = K'_{\mathtt{in}}$. Actually, the correspondence between timed and untimed configurations is maintained along the trace.

The only difficult part is when the underlying action is an input. We know that this input is performed by $a \in \mathsf{Close}(v_0)$. Let $\mathtt{in}^\star(u)$ be an input occurring in the block and let Φ' the current frame in the relaxed semantics when this action occurs and $\widehat{\Phi}$ its corresponding frame in the timed trace. By definition of the relaxed semantics, we know that there exists a recipe R such that $R\Phi'{\downarrow} = u$.

Thus, we know that $R\widehat{\Phi}\downarrow = u$. To conclude, it remains to show that the timing constraints are satisfied. We distinguish two cases:

- The input has been forged by an agent $b \in \mathsf{Close}(v_0)$. Any w used in R is either in $dom(\phi(K_{\mathtt{reset}}))$ or outputted after $L_{\mathtt{reset}}$ by an agent located at the same place as v_0. In both cases, since the global time has elapsed of $2t_0$ between $\widehat{K}$ and $\widehat{K_+}$, we know that all these w will be available at time $\widehat{t}+2t_0$ for b. Since a and b are located at the same place, we also have that this input can be done at time $\widehat{t}+2t_0$.
- The input has been forged by an agent $b \notin Close(v_0)$. In such a case, we know that $vars(R') \subseteq dom(\phi(K'_{\mathtt{reset}}))$, and thus thanks to the delay of $2t_0$ that has been applied between $\widehat{K}$ and $\widehat{K_+}$, we know that the input can be received at time $\widehat{t}+2t_0$. Indeed, b can forge the message at time $\widehat{t}+t_0$ and thus it can be received at time $\widehat{t}+2t_0$.

Note that, regarding the guarded input, the guard is trivially satisfied since the global time remains unchanged since the reset action has been performed.

Step 5: To finish, we first reduce the topology to $\mathcal{T}_{\mathsf{DH}} = (\mathcal{A}_{\mathsf{DH}}, \mathcal{M}_{\mathsf{DH}}, \mathsf{Loc}_{\mathsf{DH}}, v_0, p_0)$. Let us consider the renaming

$$\rho(a) = \begin{cases} v_0 & \text{if } a \in \mathsf{Close}(v_0) \\ p_0 & \text{if } a \notin \mathsf{Close}(v_0) \text{ and } a \in \mathcal{M}_0 \\ e_0 & \text{if } a \notin \mathsf{Close}(v_0) \text{ and } a \notin \mathcal{M}_0 \end{cases}$$

Since $\mathsf{Loc}_{\mathsf{DH}}(\rho(a)) = \mathsf{Loc}'(a)$ for any $a \in \mathcal{A}_0$, and $\rho(a) \in \mathcal{M}_{\mathsf{DH}}$ if, and only if $a \in \mathcal{M}_0$, thanks to Lemma 1, we have that:

$$K_0\rho \xrightarrow{\widehat{\mathsf{tr}_0}\rho}_{\mathcal{T}_{\mathsf{DH}}} \widehat{K}_{\mathtt{reset}}\rho \xrightarrow{L_{\mathtt{reset}}\rho;\widehat{\mathsf{tr}_1}\rho;L_{\mathtt{out}}\rho;\widehat{\mathsf{tr}_2}\rho;L_{\mathtt{in}}\rho}_{\mathcal{T}_{\mathsf{DH}}} \widehat{K}_{\mathtt{in}}\rho \xrightarrow{\widehat{\mathsf{tr}_3}\rho}_{\mathcal{T}_{\mathsf{DH}}} \widehat{K_1}\rho.$$

This execution can also be done in $\mathcal{T}^0_{\mathsf{DH}}$ since $\mathcal{T}^0_{\mathsf{DH}}$ and $\mathcal{T}_{\mathsf{DH}}$ coincide on $img(\rho)$.

To conclude, it remains to show that $K_0\rho$ is a valid initial configuration for $\overline{\mathcal{P}_{\mathsf{prox}}}$, and V_0 w.r.t. $\mathcal{T}^0_{\mathsf{DH}}$ and $\mathcal{I}_0$. Denoting Φ_0 the frame of K_0 and t_{init} the global time, we have to prove that $img(\lfloor \Phi_0\rho \rfloor_{v_0}^{t_{\mathsf{init}}}) = \emptyset$, $img(\lfloor \Phi\rho \rfloor_{e_0}^{t_{\mathsf{init}}}) = \emptyset$ and $img(\lfloor \Phi\rho \rfloor_{p_0}^{t_{\mathsf{init}}}) = \mathsf{Knows}(\mathcal{I}_0, p_0, \{v_0, p_0, e_0\})$.

First, we have that $img(\lfloor \Phi_0\rho \rfloor_{v_0}^{t_{\mathsf{init}}}) = \bigcup_{a\in\mathsf{Close}(v_0)} img(\lfloor \Phi_0 \rfloor_a^{t_{\mathsf{init}}})\rho = \emptyset$ since all these agents are honest.

Then we have that:

$$\begin{aligned} img(\lfloor \Phi_0\rho \rfloor_{e_0}^{t_{\mathsf{init}}}) &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=e_0\}} (img(\lfloor \Phi_0 \rfloor_a^{t_{\mathsf{init}}})\rho \\ &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=e_0\}} \emptyset \\ &= \emptyset \end{aligned}$$

Finally we have that:

$$\begin{aligned} img(\lfloor \Phi_0\rho \rfloor_{p_0}^{t_{\mathsf{init}}}) &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=p_0\}} (img(\lfloor \Phi_0 \rfloor_a^{t_{\mathsf{init}}})\rho \\ &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, a, \mathcal{A}_0)\rho \\ &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, \rho(a), \rho(\mathcal{A}_0)) \\ &= \textstyle\bigcup_{\{a\in\mathcal{A}_0 \ \mid \ \rho(a)=p_0\}} \mathsf{Knows}(\mathcal{I}_0, p_0, \{p_0, v_0, e_0\}) \end{aligned}$$

This allows us to conclude.

□

E Proof of Proposition 3

A process is said *initial* if it starts with an input action.

Lemma 6. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, v_0, p_0)$ be a topology, and consider an execution trace of the following form:*

$$K_0 \xrightarrow{\mathsf{tr}_0}_{\mathcal{T}} K_{\mathtt{reset}} \xrightarrow{L_{\mathtt{reset}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_1}_{\mathcal{T}} K_{\mathtt{out}} \xrightarrow{L_{\mathtt{out}}}_{\mathcal{T}} \xrightarrow{\mathsf{tr}_2}_{\mathcal{T}} K_{\mathtt{in}} \xrightarrow{L_{\mathtt{in}}}_{\mathcal{T}} K_{\mathtt{in}^+} \xrightarrow{\mathsf{tr}_3}_{\mathcal{T}} K_1$$

where $L_{\mathtt{reset}} = (v_0, \mathtt{reset}, s_0, t_{\mathtt{reset}}, \emptyset)$, $L_{\mathtt{out}} = (v_0, \mathtt{out}(v), s_0, t_{\mathtt{out}}, \mathsf{w}_0)$, and $L_{\mathtt{in}} = (v_0, \mathtt{in}^{<t_g}(u), s_0, t_{\mathtt{in}}, (b_0, t_b^0, R_0))$, and these are the only actions executed with the session identifier s_0.

Then, there exists an execution trace:

$$K'_0 \overset{\mathsf{tr}'_0}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_1}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_2}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_3}{\rightsquigarrow}_{\mathcal{T}} K'_1$$

where:

1. *K'_0 (resp. K'_1) is the untimed counterpart of K_0 (resp. K_1), i.e. $K'_i = \mathsf{untimed}(K_i)$ with $i \in \{0, 1\}$;*
2. *$L'_{\mathtt{reset}}$, $L'_{\mathtt{out}}$, and $L'_{\mathtt{in}}$ are the untimed counterpart of the annotations $L_{\mathtt{reset}}$, $L_{\mathtt{out}}$, and $L_{\mathtt{in}}$, i.e. $L'_{\mathtt{reset}} = (v_0, \mathtt{reset}, s_0, \emptyset)$, $L'_{\mathtt{out}} = (v_0, \mathtt{out}(v), s_0, \mathsf{w}_0)$, and $L'_{\mathtt{in}} = (v_0, \mathtt{in}^{<t_g}(u), s_0, (b_0, R_0))$;*
3. *for any $L = (a, \alpha, s, r)$ occurring in $\mathsf{tr}'_1.L'_{\mathtt{out}}.\mathsf{tr}'_2.L'_{\mathtt{in}}$, $2\mathsf{Dist}_{\mathcal{T}}(v_0, a) < t_g$. Moreover, when $\alpha = \mathtt{in}^\star(u)$, we have that $r = (b', R')$ for some b' and some R' such that $2\mathsf{Dist}_{\mathcal{T}}(v_0, b') < t_g$ or $vars(R') \subseteq dom(\phi(K'_{\mathtt{reset}}))$;*
4. *denoting $K'_{\mathtt{reset}} = (\mathcal{P}_{\mathtt{reset}}, \Phi_{\mathtt{reset}})$, if $\lfloor P \rfloor_a \in \mathcal{P}_{\mathtt{reset}}$ with session identifier $s_i \neq s_0$, then P is an initial process or is left unchanged after $K'_{\mathtt{reset}}$;*
5. *denoting $K'_{\mathtt{in}} = (\mathcal{P}_{\mathtt{in}}, \Phi_{\mathtt{in}})$, if $\lfloor P \rfloor_a \in \mathcal{P}_{\mathtt{in}}$ with session identifier $s_i \neq s_0$, then P is an initial process or is left unchanged after $K'_{\mathtt{in}}$.*

Proof. This proposition is an extension of Proposition 7. Indeed, applying Proposition 7 to the subtrace between $K_{\mathtt{reset}}$ and $K_{\mathtt{in}^+}$ we obtain that there exists a trace satisfying items 2 and 3

$$K'_{\mathtt{reset}} \overset{\widetilde{\mathsf{tr}_{\mathtt{reset}}}}{\rightsquigarrow}_{\mathcal{T}} \widetilde{K}_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\widetilde{\mathsf{tr}_1}}{\rightsquigarrow}_{\mathcal{T}} \widetilde{K}_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\widetilde{\mathsf{tr}_2}}{\rightsquigarrow}_{\mathcal{T}} \widetilde{K}_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\widetilde{\mathsf{tr}_{\mathtt{in}}}}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{in}^+}$$

such that $K'_{\mathtt{reset}} = \mathsf{untimed}(K_{\mathtt{reset}})$ and $K'_{\mathtt{in}^+} = \mathsf{untimed}(K_{\mathtt{in}^+})$.

Let $K'_0 = \mathsf{untimed}(K_0)$ and $K'_1 = \mathsf{untimed}(K_1)$. By definition of the relaxed semantics, we have $K'_0 \overset{\overline{\mathsf{tr}_0}}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{reset}}$ and $K'_{\mathtt{in}^+} \overset{\overline{\mathsf{tr}_3}}{\rightsquigarrow}_{\mathcal{T}} K'_1$ with $\overline{\mathsf{tr}_0}$ (resp. $\overline{\mathsf{tr}_3}$) the

untimed counterpart of tr_0 (resp. tr_3). Let $\widetilde{\mathsf{tr}_0} = \overline{\mathsf{tr}_0}.\widetilde{\mathsf{tr}_{\mathtt{reset}}}$ and $\widetilde{\mathsf{tr}_3} = \widetilde{\mathsf{tr}_{\mathtt{in}}}.\overline{\mathsf{tr}_3}$, we have that:

$$K_0' \overset{\widetilde{\mathsf{tr}_0}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K}_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_1}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K}_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_2}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K}_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widetilde{\mathsf{tr}_3}}{\rightsquigarrow_{\mathcal{T}}} K_1'$$

Moreover, we know that items 1, 2, and 3 are satisfied.

Let $\widetilde{\mathsf{tr}} = \widetilde{\mathsf{tr}_0}.L'_{\mathtt{reset}}.\widetilde{\mathsf{tr}_1}.L'_{\mathtt{out}}.\widetilde{\mathsf{tr}_2}.L'_{\mathtt{in}}.\widetilde{\mathsf{tr}_3}$. Let us prove item 4 and 5 in two steps: first we modify the trace $\widetilde{\mathsf{tr}}$ into a trace $\widehat{\mathsf{tr}}$ satisfying items 1 to 4. Then we modify $\widehat{\mathsf{tr}}$ into a trace tr' satisfying all the items (including item 5).

Step 1. In case, all the processes in $\widetilde{K}_{\mathtt{reset}}$ satisfy our requirement, we are done. Otherwise, there exists a process with session identifier s_i ($s_i \neq s_0$) which does not satisfy our requirement. Let $\widetilde{K}$ be the first configuration in the execution trace

$$\widetilde{K}_{\mathtt{reset}} \overset{L_{\mathtt{reset}}.\widetilde{\mathsf{tr}_1}.L'_{\mathtt{out}}.\widetilde{\mathsf{tr}_2}.L'_{\mathtt{in}}.\widetilde{\mathsf{tr}_3}}{\rightsquigarrow_{\mathcal{T}}} K_1'$$

such that the process with session identifier s_i is initial. If such a configuration does not exist, let $\widetilde{K} = K_1'$. Then, we consider the following set S:

$$S = \{(a, \alpha, s, r) \in \widetilde{\mathsf{tr}} \text{ before } \widetilde{K} \mid s = s_i \text{ or } (a, \alpha, s, r) \in \widetilde{\mathsf{tr}_0}\}.$$

Let $\overline{S}$ be the set of transitions β such that $\beta \notin S$ and $\beta \in \widetilde{\mathsf{tr}}$.

Claim. 1. For all $L \in S$ and $L' \in \overline{S}$, we have that $L \not\rightsquigarrow L'$.

Proof. We note $L = (a, \alpha, s, r)$ and $L' = (a', \alpha', s', r')$. By definition of S, we know that either $L \in \widetilde{\mathsf{tr}_0}$, or $s = s_i$ and $\alpha \neq \mathtt{in}^\star(u)$. Otherwise, $\widetilde{K}$ is not the first configuration in which the process with session identifier s_i is initial. In case, $L \in \widetilde{\mathsf{tr}_0}$, then we know that L' occurs before L in $\widetilde{\mathsf{tr}}$ and thus $L' \in \widetilde{\mathsf{tr}_0}$, and thus in S, leading to a contradiction. In case, $s = s_i$ and thus $\alpha \neq \mathtt{in}^\star(u)$, then we have that $L \hookrightarrow L'$ implies $L \hookrightarrow_s L'$ and thus $L' \in S$, leading to a contradiction. This concludes the proof of the claim. □

We now apply Corollary 1, and obtain a trace

$$K_0' \overset{\widetilde{\mathsf{tr}_0'}}{\rightsquigarrow_{\mathcal{T}}} \widetilde{K'}_{\mathtt{reset}} \overset{\widetilde{\mathsf{tr}'}}{\rightsquigarrow_{\mathcal{T}}} K_1'$$

such that the process with session identifier s_i is initial in $\widetilde{K'}_{\mathtt{reset}}$ or is left unchanged in the following of the trace.

We then apply the same construction on each process which does not satisfy our requirement in $\widetilde{K'}_{\mathtt{reset}}$. This construction leads to a trace $\widehat{\mathsf{tr}}$ which is:

$$K_0' \overset{\widehat{\mathsf{tr}_0}}{\rightsquigarrow_{\mathcal{T}}} \widehat{K}_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widehat{\mathsf{tr}_1}}{\rightsquigarrow_{\mathcal{T}}} \widehat{K}_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widehat{\mathsf{tr}_2}}{\rightsquigarrow_{\mathcal{T}}} \widehat{K}_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow_{\mathcal{T}}} \overset{\widehat{\mathsf{tr}_3}}{\rightsquigarrow_{\mathcal{T}}} K_1'$$

satisfying item 4. We can note that item 1, 2, and 3 are still satisfied. Indeed, we de not add any action between $L'_{\mathtt{reset}}$ and $L'_{\mathtt{in}}$.

Step 2. We now need to turn the trace into one that also satisfies item 5. In case $\widehat{K}_{\mathtt{in}}$ satisfies our requirement, we are done. Otherwise, there exists a process with session identifier s_i ($s_i \neq s_0$) which does not satisfy our requirement. First, we consider the case where the agent a_i executing session s_i is such that $\mathsf{Dist}_{\mathcal{T}}(a_i, v_0) \geq t_g$. Actually, such a case is not possible. Indeed, according to item 3, this would mean that $\widetilde{K'}_{\mathtt{reset}}$ does not satisfy our requirement. Thus, we know that the agent a_i executing session s_i is such that $\mathsf{Dist}_{\mathcal{T}}(a_i, v_0) < t_g$. In such a case, we proceed as in Step 1. Note that actions that will be added between $L'_{\mathtt{reset}}$ and $L'_{\mathtt{in}}$ are only outputs or τ actions executed by agents in the neighbourhood of v_0, i.e. such that $\mathsf{Dist}_{\mathcal{T}}(a, v_0) < t_g$. Thus, item 3 is still satisfied. □

Proposition 3. *Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}_0, v_0, p_0)$ be a topology, $\mathcal{P}_{\mathsf{prox}}$ a protocol, $t_0 \in \mathbb{R}_+$, $\mathcal{I}_0$ a template, and $V_0(z_0, z_1)$ a parametrised process of the form:*

$$\mathsf{block}_1 \,.\, \mathtt{reset} \,.\, \mathtt{out}(m) \,.\, \mathtt{in}^{<t}(x) \,.\, \mathsf{block}_2 \,.\, \mathsf{end}(z_0, z_1) \quad \textit{with } t \leq 2 \times t_0$$

Let K_0 be a valid initial configuration for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$. If K_0 admits an attack w.r.t. t_0-proximity in $\mathcal{T}$, then we have that:

$$\mathcal{F}(\mathcal{T}_0, \overline{\mathcal{P}_{\mathsf{prox}}}, V_0, \mathcal{I}_0, t_0) \overset{\mathsf{tr}}{\Longrightarrow} (\{2 : \mathsf{end}(v_0, p_0)\} \uplus \mathcal{P}; \phi; 2).$$

Moreover, in case there is no $a \in \mathcal{M}_0$ such that $\mathsf{Dist}_{\mathcal{T}_0}(a, v_0) < t_0$, we have that for any $\mathtt{in}(u)$ occurring in tr during phase 1, the underlying recipe R is either of the form w, or only uses handles ouputted in phase 0.

Proof. Let $\mathcal{T} = (\mathcal{A}_0, \mathcal{M}_0, \mathsf{Loc}, p_0, v_0)$ be a topology, $\mathcal{P}_{\mathsf{prox}}$ a protocol, $\mathcal{I}_0$ a template and V_0 a parametrised process of the right form. Let K_0 be a valid initial configuration for the protocol $\mathcal{P}_{\mathsf{prox}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$ such that

$$K_0 \xrightarrow{\mathsf{tr}}_{\mathcal{T}} (\{ \lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_{v_0}} \} \uplus \mathcal{P}; \Phi; t) = K_1 \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Let $\overline{K_0}$ the configuration obtained from K_0 by removing `reset` instructions, and replacing guarded inputs by simple inputs for processes other than V_0. We have that $\overline{K_0}$ is a valid initial configuration for $\overline{\mathcal{P}_{\mathsf{prox}}}$ and V_0 w.r.t. $\mathcal{T}$ and $\mathcal{I}_0$. Moreover, the trace tr is still executable (up to some `reset` instructions) and leads to $\overline{K_1}$, i.e. K_1 in which `reset` instructions have been removed and guarded inputs have been replaced by simple inputs.

Therefore, we have that:

$$\overline{K_0} \xrightarrow{\overline{\mathsf{tr}}}_{\mathcal{T}} (\{ \lfloor \mathsf{end}(v_0, p_0) \rfloor_{v_0}^{t_{v_0}} \} \uplus \overline{\mathcal{P}}; \Phi; t) = \overline{K_1} \text{ with } \mathsf{Dist}_{\mathcal{T}}(v_0, p_0) \geq t_0.$$

Due to the specific form of V_0 we know that $\overline{\mathsf{tr}}$ has the right form to apply Lemma 6. Thus, we know that there exists a trace tr' such that:

$$K'_0 \overset{\mathsf{tr}'_0}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{reset}} \overset{L'_{\mathtt{reset}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_1}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{out}} \overset{L'_{\mathtt{out}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_2}{\rightsquigarrow}_{\mathcal{T}} K'_{\mathtt{in}} \overset{L'_{\mathtt{in}}}{\rightsquigarrow}_{\mathcal{T}} \overset{\mathsf{tr}'_3}{\rightsquigarrow}_{\mathcal{T}} K'_1$$

with:

1. K'_0 (resp. K'_1) is the untimed counterpart of $\overline{K_0}$ (resp. $\overline{K_1}$), i.e. $K'_i = \mathsf{untimed}(\overline{K_i})$ with $i \in \{0,1\}$;
2. $L'_{\mathtt{reset}}$, $L'_{\mathtt{out}}$, and $L'_{\mathtt{in}}$ correspond to the $\mathtt{reset.out}(u).\mathtt{in}^{<t}(x)$ block occurring in V_0;
3. for any $L = (a, \alpha, s, r)$ occurring in $\mathsf{tr}'_1.L'_{\mathtt{out}}.\mathsf{tr}'_2.L'_{\mathtt{in}}$, $2\mathsf{Dist}_{\mathcal{T}}(v_0, a) < t$. Moreover, when $\alpha = \mathtt{in}^{\star}(u)$, we have that $r = (b', R')$ for some b' and some R' such that $2\mathsf{Dist}_{\mathcal{T}}(v_0, b') < t$ or $vars(R') \subseteq dom(\phi(K'_{\mathtt{reset}}))$;
4. denoting $K'_{\mathtt{reset}} = (\mathcal{P}_{\mathtt{reset}}, \Phi_{\mathtt{reset}})$, if $\lfloor P \rfloor_a \in \mathcal{P}_{\mathtt{reset}}$ with session identifier $s_i \neq s_0$, then P is an initial process or is left unchanged after $K'_{\mathtt{reset}}$;
5. denoting $K'_{\mathtt{in}} = (\mathcal{P}_{\mathtt{in}}, \Phi_{\mathtt{in}})$, if $\lfloor P \rfloor_a \in \mathcal{P}_{\mathtt{in}}$ with session identifier $s_i \neq s_0$, then P is an initial process or is left unchanged after $K'_{\mathtt{in}}$.

To conclude, it remains to show that such a trace tr' can actually be mimicked from the configuration $\mathcal{F}(\mathcal{T}, \overline{\mathcal{P}_{\mathsf{prox}}}, V_0, \mathcal{I}_0, t_0)$. Actually, some τ actions (REP rule) have to be added, $L'_{\mathtt{reset}}$ is replaced by $\mathsf{phase}\ 1$, and $L'_{\mathtt{in}}$ is followed by a $\mathsf{phase}\ 2$ action.

Let $\lfloor P \rfloor_a^t$ be a process occurring in $\overline{K_0}$, we know that $P = R(a, a_1, \ldots, a_n)$ with $R(z_0, z_1, \ldots, z_n)$ a role of $\overline{\mathcal{P}_{\mathsf{prox}}}$. We know consider all the actions performed by this process along the trace tr', and in particular, we pay attention on the slicing of all these actions w.r.t. $L'_{\mathtt{reset}}$ and $L'_{\mathtt{in}}$. This gives us the corresponding process that we have to consider in our translation, so that it will be able to mimic all the actions of $\lfloor P \rfloor_a^t$. Items 4 and 5 allow one to ensure that our slicing (just before the inputs) is indeed sufficient. Our transformation $\mathcal{F}^{\geq}$ also forbid actions to be executed during phase 1, and this is justified by our item 3.

Finally, it remains to establish that in case there is no $a \in \mathcal{M}_0$ such that $\mathsf{Dist}_{\mathcal{T}}(a, v_0) < t_0$, we have that for any $\mathtt{in}(u)$ occurring in the trace during phase 1, the underlying recipe R is either of the form w, or only uses handles outputted in phase 0.

To establish this, we rely on item 3. We know that for any $\mathtt{in}(u)$ (with annotation $(a, \mathtt{in}^{\star}(u), s, r)$) occurring in the trace during phase 1, we have that $r = (b', R')$ for some b' and some R' such that $2\mathsf{Dist}_{\mathcal{T}}(v_0, b') < t$ or R' only uses handles outputted in phase 0. Since, by hypothesis, we know that there is no $a \in \mathcal{M}_0$ such that $\mathsf{Dist}_{\mathcal{T}}(a, v_0) < t_0$, thus we know that there is no $a \in \mathcal{M}_0$ such that $2\mathsf{Dist}_{\mathcal{T}}(a, v_0) < 2t_0$, and thus there is no $a \in \mathcal{M}_0$ such that $2\mathsf{Dist}_{\mathcal{T}}(a, v_0) < t$ (since $t \leq 2 \times t_0$). Thus, we deduce that either b' is honest or R' only uses handles outputted in phase 0, which implies that R' is either of the form w, or only uses handles outputted in phase 0. □