Distance-Bounding Protocols: Verification without Time and Location[*]

Sjouke Mauw[1,2] Zach Smith[1] Jorge Toro-Pozo[1] Rolando Trujillo-Rasua[2]

[1]CSC/[2]SnT, University of Luxembourg
6, avenue de la Fonte, L-4364 Esch-sur-Alzette, Luxembourg
{sjouke.mauw, zach.smith, jorge.toro, rolando.trujillo}@uni.lu

Abstract

Distance-bounding protocols are cryptographic protocols that securely establish an upper bound on the physical distance between the participants. Existing symbolic verification frameworks for distance-bounding protocols consider timestamps and the location of agents. In this work we introduce a causality-based characterization of secure distance-bounding that discards the notions of time and location. This allows us to verify the correctness of distance-bounding protocols with standard protocol verification tools. That is to say, we provide the first fully automated verification framework for distance-bounding protocols. By using our framework, we confirmed known vulnerabilities in a number of protocols and discovered unreported attacks against two recently published protocols.

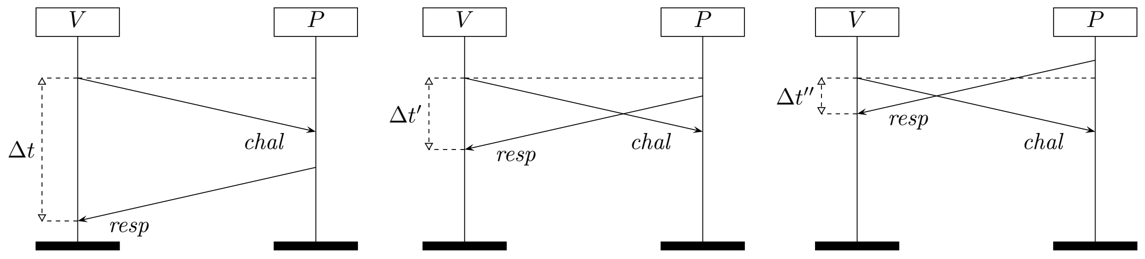
1 Introduction

Contactless systems are gaining more and more popularity nowadays. An increasing number of applications, including ticketing, access control, e-passports, tracking services, and mobile payments, make use of contactless communication technologies such as RFID and NFC. However, contactless communication is known to be vulnerable to *relay attacks* [1]: a man-in-the-middle attack where an adversary relays the verbatim messages that are being exchanged through the network.

Relay attacks are mostly used to break communication protocols with a bounded read range, such as smartcards (2-10 cm) or car keys (10-100 m). By simply relaying, an adversary is able to establish a long-range communication between two contactless tokens, which otherwise wouldn't be possible. This has been used, for example, by Francillon et al. [2] to break the passive keyless entry system of various modern cars.

To face relay attacks, Desmedt et al. [3, 4] introduced the notion of *distance-bounding protocols*, and the first such protocol was designed by Brands and Chaum [5]. Distance-bounding protocols use the round-trip time (RTT) of one or more challenge/response rounds to provide an upper bound on the prover-to-verifier distance (see Figure 1a). Through this

[*]Accepted for IEEE S&P 2016. For reference, use the version at IEEE Computer Society's Digital Library, DOI: 0.1109/SP.2018.00001. This work was supported by the Luxembourg National Research Fund (FNR), under the grants AFR-PhD-10188265 and C15-IS-10428112.



(a) (b) (c)

Figure 1: Three timing scenarios of a challenge/response round.

scheme, security verification translates into the validity of the actual prover-to-verifier distance in comparison with the RTTs. More precisely, in a secure distance-bounding protocol, if the prover-to-verifier distance is d and the RTT is Δt, then it must hold that $d \leq \frac{1}{2}\Delta t \cdot \mathsf{c}$, where c denotes the maximum network transmission speed (for radio-waves, this is the speed of light). This intuition is supported by the physical fact that no message can be transmitted at a speed higher than c.

In the context of distance-bounding protocols, their security has traditionally been verified over the years, by accounting for their resistance to three types of attack: mafia fraud [1], distance fraud [6], and terrorist fraud [6]. Resistance is measured in terms of probability of success of the adversary in a given adversary model [7, 8, 9]. However, this probabilistic analysis based on attack-resistance does not seem to be a promising verification scheme, as new attacks might be discovered in the future.

A clear and convincing proof of the flaws of the attack-based security analysis is the work by Cremers et al. [10]. In this work, the authors prove several protocols to be vulnerable to *distance-hijacking* attacks while they were previously considered secure as they resisted the then-existing attack types (mafia, distance and terrorist frauds). An important observation is that, like previous authors on distance bounding, Cremers et al. assumed a Dolev-Yao [11] adversary, so they did not introduce stronger adversary models to define their new type of attack.

Unfortunately, although the desired properties of a distance-bounding protocol can be precisely defined in current security models, it is not so straightforward to verify that a given protocol satisfies these properties. On the one hand, computational models [8, 12] typically lead to manual and complex security proofs. On the other hand, symbolic models [13, 10] rely on using adapted versions of higher-order theorem-proving tools such as Isabelle/HOL [14], which require a high degree of user intervention. This means that verifying the security of a distance-bounding protocol in the existing symbolic models requires not only a considerable amount of expertise, but also a significant time investment.

By comparison, well-established verification tools, such as Tamarin [15], ProVerif [16] and Scyther [17], are able to verify traditional authentication properties in a straightforward and rapid way. These tools handle time as a discrete ordering of events, therefore verifying protocols with the notion of continuous time becomes difficult.

In this paper we argue that the notions of time and location are indeed *not* needed to specify and verify the security of distance-bounding protocols. Surprisingly enough, such protocols can be verified by considering the causal order of events in protocol traces, similarly to authentication properties like aliveness and synchronization [18]. The intuition behind this observation is illustrated in Figure 1.

Figure 1a shows a regular challenge/response round, in which prover P can only respond to verifier V's challenge after having received the challenge. Therefore, $\frac{1}{2}\mathsf{c} \cdot \Delta t$ determines an upper bound on the distance d between V and P. Now, suppose that, due to a vulnerability of the protocol, P is able to predict the appropriate response before having received the challenge (Figure 1b). This means that he will be able to send his response "too early", leading to a shorter round-trip time $\Delta t' < \Delta t$ and thus to a smaller and incorrect distance calculated by V. Thus, if the protocol is insecure because P can preempt the response, P has sufficient knowledge to create the response before reception of the challenge. Now our main observation is that (assuming that there is no other causal relation between sending the challenge and P's knowledge), P could even have sent the response *before V sent the challenge* (Figure 1c). From a causal point of view, this means that if there is a trace in which P sends its response before P receives the challenge, there must also be a trace in which P sends the response before V sends the challenge. Hence, a flaw in the protocol translates into such a wrongly ordered trace, which can be discovered through an analysis that does not consider time.

In the remainder of this paper, we will make this high-level intuition precise in the following way:

- First we introduce a security model, based on Basin et al. [19, 13] and formally define the notion of *secure distance-bounding* using time and location (Section 3).
- Then, in Section 4, we analyse the semantic domain and formulate a number of basic properties that provide a sufficient characterization of the semantics to prove our main result. The purpose of this step is to make our result independent of the particular time/location semantics used.
- Next, we formulate our notion of *causality-based secure distance-bounding*, which does not refer to time and location, and we prove it equivalent to the previously defined notion of *secure distance-bounding* (Section 5).
- In order to validate our results, we demonstrate an implementation of causality-based secure distance-bounding in Tamarin [15] and use it to perform a large-scale analysis of published protocols (Section 6). Our analysis results coincide with previous formal analyses, such as the report by Cremers et al. [10]. In addition, we uncover previously unreported vulnerabilities on recently published protocols.

2 Background

Distance-Bounding Protocols. The first distance-bounding protocol was designed by Brands and Chaum [5] and it is composed of three phases. The *slow phase* (a.k.a. initial phase, setup phase) is where the parties agree on the parameters of the session, such as nonces. Then the *fast phase* (a.k.a. critical phase, distance-bounding phase, timed phase) is executed, consisting of a number of challenge/responses rounds, where the verifier measures the round-trip times. Finally, a *verification phase* (a.k.a. final phase, authentication phase) takes place, in which the verifier makes a decision on whether the prover successfully passed

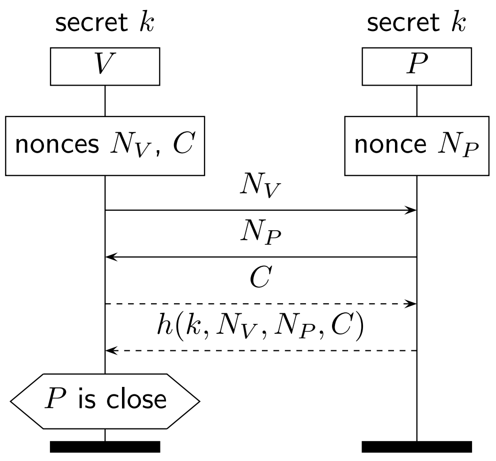


Figure 2: A representation of Hancke and Kuhn's protocol.

the protocol. This is done by checking the correctness of all round-trip times and the prover's proof of knowledge of a valid signature.

Another well-known distance-bounding protocol was proposed by Hancke and Kuhn in [20]. An abstraction of this protocol is shown in Figure 2. The first two messages compose the initial phase of the protocol, where the verifier V sends his nonce N_V to the prover P who replies back with his nonce N_P. Then the fast phase starts (represented by dashed arrows) with V sending his challenge C to P whose response is $h(k, N_V, N_P, C)$, where k is the shared secret key between V and P and h is an irreversible cryptographic function. The verification phase is represented by V's claim that "P is close". The protocol seems to be secure, as for an attacker (who could be an untrusted prover) to pass the protocol, he must know either the verifier's challenge in advance or the shared secret key between the verifier and the intended prover. However, due to the particular choice of h, a mafia-fraud attacker successfully passes the protocol with a non-negligible probability of $(3/4)^{|C|}$ (see [20] for further details).

One of the main differences between Brands and Chaum's protocol and Hancke and Kuhn's protocol is the following. In the former, the fast phase messages do not rely on long-term secret keys whereas in the latter protocol, such a reliance does exist. Various protocols have been proposed following this characteristic of Brands and Chaum's approach, e.g. [21, 22, 23, 24, 25] whilst others employ Hancke and Kuhn's design, such as [26, 27, 28, 29, 30].

Attacks on Distance-Bounding Protocols. Although distance-bounding protocols solved the problem of relay attacks to some extent, more sophisticated attacks have emerged, such as *mafia fraud*, *distance fraud*, *terrorist fraud* and *distance hijacking*.

Mafia-fraud attacks were introduced in [1], in which a dishonest agent I uses an honest prover P to provide a verifier V with a false upper bound on the distance between V and P. Some authors consider mafia-fraud attacks to be the same as relay attacks. Others, however, classify mafia-fraud attackers stronger than relay attackers by assuming that the former can manipulate/modify the messages, rather than simply relaying them.

A distance-fraud attacker [6] is a dishonest prover I whose goal is to provide a verifier V with a false upper bound on V's distance to I. In particular, for this type of attack, I does not use any other prover to perform his attack.

More sophisticated attacks are *terrorist fraud* and *distance hijacking.* Terrorist-fraud attacks were first discussed in [6] in which the attacker prover I cheats on the upper bound on the distance between a verifier V and a dishonest prover P, without learning P's secret key material. Distance hijacking was introduced by Cremers at al. in [10], in which a dishonest prover I makes use of honest provers in order to provide a verifier V with a false upper bound on the distance between V and I.

Probabilistic Security Analysis. The work by Avoine et al. [7] introduces a framework that explores the adversary's capabilities and strategies and the influence of provers' abilities to tamper with their devices. New concepts in the distance-bounding field are introduced such as black-box and white-box models.

The concepts sketched in [7] were soon formulated in computational models. For example, Dürholz et al. formalized the classical frauds (except for distance hijacking) by using an adversary model that does not allow for corrupted verifiers [8]. Boureanu, Mitrokotsa, and Vaudenay introduced a more general model [12] by allowing adversaries to interact with multiple provers and verifiers, hence capturing distance hijacking [10].

Mauw, Toro-Pozo, and Trujillo-Rasua [28, 31] developed a probabilistic analysis of the security of a class of distance-bounding protocols in terms of mafia fraud. This class includes distance-bounding protocols that do not have a final verification phase and are based on precomputation (e.g. [20, 26, 32, 33, 34, 28, 30, 27]). They proposed a set-of-automata representation of protocols that allows the analyst to generically compute the success probability of mafia-fraud attacks.

Symbolic Security Analysis. Meadows et al. [23] proposed a formal framework to verify distance-bounding protocols. Their approach does not particularly deal with multi-prover scenarios, therefore neither distance-hijacking nor terrorist-fraud attacks would be detected.

The first formal framework for distance-bounding protocols with multi-prover scenarios was proposed by Malladi et al. [35], along with a software tool. They analyse the signature-based Brands and Chaum's protocol and find an attack in which an adversary who is not in the vicinity of the verifier still passes the protocol. They call this attack the *farther adversary* scenario. Moreover, to solve the security issue they found, they observed that including the prover's identity in the signature would make the protocol no longer vulnerable to farther adversary attacks.

Basin et al. [19, 13] introduced a simple yet powerful formal approach for distance-bounding verification. Their model captures dishonest prover behaviors and, by extension, distance-fraud and distance-hijacking attacks, of which the latter was referred to as *impersonation attacks.* Their implementation of the formalization is written in the higher-order logic theorem prover Isabelle/HOL [14]. Similarly to Malladi et al. [35], they prove that the signature-based Brands and Chaum's protocol can be fixed by explicitly adding the prover's identity to the responses in the fast phase.

In [10], Cremers et al. extended Basin et al's model to capture bit-level message manipulation on wireless networks, introduced as *overshadowing* in [36]. Supported by this, they proved that including the prover's identity (neither by XOR-ing it with the challenge responses nor by using secure channels) in Brands and Chaum's protocol does not solve its vulnerability to distance hijacking.

It is still an open problem how to model terrorist fraud in a symbolic security model. Originally, a terrorist-fraud attack consisted in a far-away dishonest prover passing the distance-bounding protocol with the help of the adversary, but without leaking the prover's long-term key [6]. Many attempts to formalize this intuition have been made in computational models [8, 37, 38, 24]. Yet, there seems to be no agreement on the appropriate definition.

3 A Security Model Based on Time and Location

In this section we describe the formalism of Basin et al. [19, 13] which is the basis for our work. The formalism employs logic theories to handle inductively-defined sets of traces that represent the protocol's executions. It considers execution traces that consist of a sequence of timed-events, e.g. denoting the sending and reception of messages, where the timestamps represent the point in time at which the events occurred.

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

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

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

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

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

The set $\mathcal{B} = \{sk, pk, sh, \_^{-1}\} \subseteq \mathcal{F}$ is the set of *basic functions* and its functions are assumed to satisfy that $sh(A,B) = sh(B,A)$, $pk(A)^{-1} = sk(A)$ and $sk(A)^{-1} = pk(A)$; for all $A, B \in \mathsf{Agent}$. In addition, we assume that $k \notin \{pk(A), sk(A)\}$ implies $k^{-1} = k$; for all $k \in \mathsf{Msg}$ and $A \in \mathsf{Agent}$. These assumptions represent the properties for symmetric and asymmetric encryption/decryption.

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

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

$$\frac{m \in \mathit{init}(A)}{m \in dm_A(\alpha)} \qquad \frac{(t, \mathsf{recv}_A(m)) \in \alpha}{m \in dm_A(\alpha)} \qquad \frac{\begin{array}{c} m_1 \in dm_A(\alpha) \\ m_2 \in dm_A(\alpha) \end{array}}{(m_1, m_2) \in dm_A(\alpha)} \qquad \frac{\begin{array}{c} m \in dm_A(\alpha) \\ f \in \mathcal{F} \setminus \mathcal{B} \end{array}}{f(m) \in dm_A(\alpha)}$$

$$\frac{\begin{array}{c} (m_1, m_2) \in dm_A(\alpha) \\ i \in \{1, 2\} \end{array}}{m_i \in dm_A(\alpha)} \qquad \frac{\begin{array}{c} m \in dm_A(\alpha) \\ k \in dm_A(\alpha) \end{array}}{\{m\}_k \in dm_A(\alpha)} \qquad \frac{\begin{array}{c} \{m\}_k \in dm_A(\alpha) \\ k^{-1} \in dm_A(\alpha) \end{array}}{m \in dm_A(\alpha)}$$

Figure 3: Rules for message deduction.

Given messages m_1 and m_2, and agents A and B, $\mathsf{send}_A(m_1)[m_2]$ indicates that A has sent the message m_1 and updated the agent's local state with the message m_2, and $\mathsf{recv}_A(m_1)$ means that A has received m_1. In the original model, claiming events have the form $\mathsf{claim}_A(B, d)$, where $d \in \mathbb{R}$ is a distance value. This allows an agent A to claim that another agent B is within a radius of length d, which is computed based on the round-trip time of a message exchange. We will make the message exchange explicit, and use $\mathsf{claim}_A(B, e_1, e_2)$ where e_1 and e_2 are the events used to compute the round-trip time and, by extension, the distance bound d.

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

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

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

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

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

Given an agent A and a trace α, $dm_A(\alpha)$ denotes the set of all *deducible messages* from a trace α. This set is inductively defined by the rules in Figure 3.

Network and Intruder. For a given protocol $\mathcal{P}$, the set of possible traces $\mathsf{Tr}(\mathcal{P})$ is inductively defined by the Start rule (`Start`), the Intruder rule (`Int`), the Network rule (`Net`) and the rules specifying the protocol. The Start, Intruder and Network rules are depicted in Figure 4.

The rules make use of the function $\mathit{maxt} \colon (\mathbb{R} \times \mathsf{Ev})^* \to \mathbb{R}$, defined as $\mathit{maxt}(\alpha) = \max_{(t,e) \in \alpha}\{t\}$, yields the latest time at which an event of α occurred. The expression

$$\frac{}{\epsilon \in \mathsf{Tr}(\mathcal{P})}\ \texttt{Start}$$

$$\frac{\begin{array}{cc}\alpha \in \mathsf{Tr}(\mathcal{P}) & I \in \mathsf{Dishonest} \\ t \geq maxt(\alpha) & m \in dm_I(\alpha)\end{array}}{\alpha \cdot (t, \mathsf{send}_I(m)[]) \in \mathsf{Tr}(\mathcal{P})}\ \texttt{Int} \qquad \frac{\begin{array}{c}\alpha \in \mathsf{Tr}(\mathcal{P}) \quad t \geq maxt(\alpha) \\ (t', \mathsf{send}_A(m)[s]) \in \alpha \\ t \geq t' + d(A,B)/\mathsf{c}\end{array}}{\alpha \cdot (t, \mathsf{recv}_B(m)) \in \mathsf{Tr}(\mathcal{P})}\ \texttt{Net}$$

Figure 4: Start, Intruder and Network rules.

$d(A, B)$ gives the distance between two agents A and B based on an uninterpreted function $l\colon \mathsf{Agent} \to \mathbb{R}^3$, which associates each agent to a location in the real coordinate space $\mathbb{R}^3$. It is worth remarking that this interpretation of location assumes that agents are static, including dishonest agents.

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

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

Protocol Specification. A protocol is specified by a set of rules similar to the rules in Figure 4. Two syntactic restrictions (whose semantic interpretations will be given in Section 4.1) are applied:

- Neither the premises nor the conclusion of a protocol rule contain references to dishonest agents. This means that the behavior of dishonest agents is fully specified by the intruder rule.
- The premise of a protocol rule cannot contain events whose actors are not the same as the actor of the event in the premise of the rule. That is to say, agents are unaware of what other agents do. They can interact exclusively through the network rule.

Example 1 (Hancke and Kuhn's protocol)**.** *Figure 5 shows the formalization of Hancke and Kuhn's protocol [20] (see the representation in Figure 2). The first four rules in Figure 5 correspond to the four transmissions that take place in the protocol. The receiving events are derived from the network rule. The last rule from Figure 5 refers to the claim event for the property* secure distance-bounding *represented as "P is close" in Figure 2.*

The function used: $(\mathbb{R} \times \mathsf{Ev})^* \to \mathcal{P}(\mathsf{Msg})$ *defined as* $used(\alpha) = \bigcup_{(t,e)\in\alpha} subt(cont(e))$, *is utilized to make sure that newly generated nonces are fresh, where* $subt\colon \mathsf{Msg} \to \mathcal{P}(\mathsf{Msg})$ *indicates the set of atomic messages that are sub-terms of a given message and* $cont\colon \mathsf{Ev} \to$

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

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

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

$$\frac{\alpha \in \mathsf{Tr}(\mathcal{P}) \quad P \in \mathsf{Honest} \quad t \geq maxt(\alpha) \\ (t', \mathsf{send}_P(N_P)\,[N_V]) \in \alpha \quad (t'', \mathsf{recv}_P(C)) \in \alpha}{\alpha \cdot (t, \mathsf{send}_P(h(sh(V,P), N_V, N_P, C))\,[]) \in \mathsf{Tr}(\mathcal{P})}$$

$$\frac{\alpha \in \mathsf{Tr}(\mathcal{P}) \quad V \in \mathsf{Honest} \quad tw \geq maxt(\alpha) \\ u = \mathsf{send}_V(C)\,[N_V, N_P] \\ v = \mathsf{recv}_V(h(sh(V,P), N_V, N_P, C)) \\ (tu, u) \in \alpha \quad (tv, v) \in \alpha}{\alpha \cdot (tw, \mathsf{claim}_V(P, u, v)) \in \mathsf{Tr}(\mathcal{P})}$$

Figure 5: Formalization of Hancke and Kuhn's protocol.

Msg *gives us the content of a given event. The function subt is recursively defined as follows.*

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

Example 1 also illustrates the purpose of the information in square brackets at the end of the send actions. In this case, it is implicitly used to define the notion of a session, by extending the send actions with the random nonces from that session. Further, it is used to specify in which order the events of a session will have to be executed.

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

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

$$\approx = \{(A, A) \mid A \in \mathsf{Honest}\} \cup \mathsf{Dishonest} \times \mathsf{Dishonest}.$$

We use $A \not\approx B$ to indicate that $(A, B) \notin \approx$. By considering the relation $\approx$, we provide next a formal definition of *secure distance-bounding.*

Definition 1 (Secure distance-bounding)**.** *A protocol $\mathcal{P}$ satisfies* secure distance-bounding *if and only if:*

$$\begin{aligned}
&\forall \alpha \in \mathsf{Tr}(\mathcal{P}), V, P \in \mathsf{Agent}, u, v, w \in \mathsf{Ev}, tw \in \mathbb{R}: \\
&\quad (tw, w) \in \alpha \wedge w = \mathsf{claim}_V(P, u, v) \implies \\
&\quad\quad \exists tu, tv \in \mathbb{R}, P' \in actor(\alpha): \\
&\quad\quad\quad (tu, u) \in \alpha \wedge (tv, v) \in \alpha \wedge P \approx P' \wedge d(V, P') \leq \frac{\mathsf{c}}{2}(tv - tu). \qquad (1)
\end{aligned}$$

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

Our definition of secure distance-bounding slightly differs from the original one provided by Basin et al., but the difference is merely notational, allowing us to cleanly formulate our main result in Section 5. Note that claim events are formulated in such a way that they relate to a single challenge/response pair. Thus, similar to Basin et al's approach, we will need to include several claim events if the fast phase cannot be abstracted to a single challenge/response pair.

4 The Semantic Domain

An important characteristic of Basin et al's approach, as presented in the previous section, is that security protocols are specified using the same type of derivation rules as used for the definition of the general semantics of the system. Consequently, protocol specifications are much more liberal than in comparable formal approaches that define a domain specific language for the definition of protocols. Alternative approaches, like the one by Cremers and Mauw [18] provide a dedicated protocol specification language and impose syntactical or semantical constraints to prevent users from specifying meaningless or simply undesired protocols.

An example of a protocol rule that may be considered undesirable is the one in Figure 6. It specifies that after reception of the message Hello at time t, agent A sends a message Hi back at time $t - 1$. This is clearly an infringement of a time consistency property, because it leads to the trace $(1, \mathsf{recv}_A(\mathsf{Hello})) \cdot (0, \mathsf{send}_A(\mathsf{Hi})\,[])$.

The solution proposed by Basin et al. is to consider only those traces that have non-decreasing timestamps for subsequent events. In our approach we will take this line of reasoning one step further, in that we will define a number of assumptions that a proper semantics should satisfy and that are sufficient to derive our main result. We will argue that

$$\frac{\alpha \in \mathsf{Tr}(\mathcal{P}) \qquad A \in \mathsf{Honest} \qquad \mathsf{Hello}, \mathsf{Hi} \in \mathsf{Const} \qquad (t, \mathsf{recv}_A(\mathsf{Hello})) \in \alpha}{\alpha \cdot (t-1, \mathsf{send}_A(\mathsf{Hi})\,[\,]) \in \mathsf{Tr}(\mathcal{P})}$$

Figure 6: A protocol rule that leads to incorrect traces.

these properties are valid for the semantics from the previous section, under the assumption of a class of "reasonable" protocol specifications.

4.1 Basic Properties of the Semantics

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

Property 1 (Time consistency). *A protocol $\mathcal{P}$ satisfies* time consistency *if for every trace $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in \mathsf{Tr}(\mathcal{P})$, it holds that $t_1 \leq \cdots \leq t_n$.*

The second property that we consider is *speed-of-light consistency.* It states that all traces satisfy the restrictions of the speed of light. In particular, this means that the time between the sending of a message by agent A and the reception of this message by agent B must be equal to or larger than the distance between the two agents divided by the speed of light.

Because this definition requires the correspondence between a send event and its related receive event, we define the relation $\rightsquigarrow \subseteq \mathsf{Send} \times \mathsf{Recv}$ as follows:

$$\rightsquigarrow = \{(e, e') \in \mathsf{Send} \times \mathsf{Recv} \mid \mathit{cont}(e) = \mathit{cont}(e')\}.$$

The relation $\rightsquigarrow$ defines whether an event e' is a receive event that could have occurred as consequence of the send event e. As followed from its formulation, $\rightsquigarrow$ is not a one-to-one relation. This lines up with the fact that it does not need to be the case that there is a unique send event that triggers a given receive event. In the semantics above, the relation $\rightsquigarrow$ can be easily derived from the application of the Network rule in Figure 4.

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

Even though we define Properties 1 and 2 for protocols, we will also use them in relation to traces. Thus we will talk about time consistency and speed-of-light consistency of a given trace, with the obvious interpretation.

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

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

Property 3 (Prefix-closure). *A protocol $\mathcal{P}$ is* prefix-closed *if for every $\gamma = \sigma \cdot e \in \pi(\mathsf{Tr}(\mathcal{P}))$, it holds that $\sigma \in \pi(\mathsf{Tr}(\mathcal{P}))$.*

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

Property 4 (Time-unawareness)**.** *A protocol* $\mathcal{P}$ *is* time-unaware *if for every trace* $\alpha \in \mathsf{Tr}(\mathcal{P})$ *the following holds: for all time consistent and speed-of-light consistent traces* $\beta \in (\mathbb{R} \times \mathsf{Ev})^*$*,* $\alpha \sim \beta$ *implies* $\beta \in \mathsf{Tr}(\mathcal{P})$*.*

As mentioned in Section 3, different agents only interact through the network via sending and receiving events. As a consequence, a non-receive action can only be triggered by the actor agent's own preceding actions and another agent's actions in between can be disregarded or delayed. This leads to the fifth property, *locally-enabled events*. We use untimed events in order to easily express that the resulting trace $\sigma \cdot e'$ might require a re-timing of event e'.

Property 5 (Locally-enabled events)**.** *A protocol* $\mathcal{P}$ *satisfies* locally-enabled events *if for every* $\gamma = \sigma \cdot e \cdot e' \in \pi(\mathsf{Tr}(\mathcal{P}))$ *such that* $e' \notin \mathsf{Recv}$ *and* $\mathit{actor}(e) \neq \mathit{actor}(e')$*, it holds that* $\sigma \cdot e' \in \pi(\mathsf{Tr}(\mathcal{P}))$*.*

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

Property 6 (Transmission-enabled events)**.** *A protocol* $\mathcal{P}$ *satisfies* transmission-enabled events *if for every* $\gamma = \sigma \cdot e \in \pi(\mathsf{Tr}(\mathcal{P}))$ *and every* $e' \in \mathsf{Recv}$ *such that* $e \rightsquigarrow e'$*, it holds that* $\gamma \cdot e' \in \pi(\mathsf{Tr}(\mathcal{P}))$*.*

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

Property 7 (Substitution-closure)**.** *A protocol* $\mathcal{P}$ *is* substitution-closed *if for every* $\sigma \in \pi(\mathsf{Tr}(\mathcal{P}))$ *and every* $A, B \in \mathsf{Agent}$ *such that* $\{A, B\} \subseteq \mathsf{Honest}$ *or* $\{A, B\} \subseteq \mathsf{Dishonest}$*, it holds that* $\sigma[A \mapsto B] \in \pi(\mathsf{Tr}(\mathcal{P}))$*.*

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

4.2 Validity of the Properties

As stated in the beginning of the current section, the mechanism for specifying protocols is too liberal to ensure the well-formedness properties. Therefore, we use a restricted format for protocol rules inspired by the example specification of Hancke and Kuhn's protocol from Figure 5. The restricted format is specified by the rule prototype in Figure 7. We additionally require that $p + q > 0$, $A = \mathit{actor}(e) = \mathit{actor}(e_1) = \mathit{actor}(e_2) = \cdots = \mathit{actor}(e_q)$, $e \notin \mathsf{Recv}$ and none of the premises prem_i involve any of the timestamps t_j or t. Even though the

$$\frac{\begin{array}{c}\alpha \in \mathsf{Tr}\,(\mathcal{P}) \qquad A \in \mathsf{Honest} \qquad t \geq maxt(\alpha) \\ prem_1 \quad prem_2 \quad \cdots \quad prem_p \\ (t_1, e_1) \in \alpha \quad (t_2, e_2) \in \alpha \quad \cdots \quad (t_q, e_q) \in \alpha\end{array}}{\alpha \cdot (t, e) \in \mathsf{Tr}\,(\mathcal{P})}$$

Figure 7: Prototype of rules that lead to well-formed protocols.

protocol format is restricted with respect to the liberal format specified by Basin et al., we conjecture that it is sufficiently expressive to specify all relevant protocols from literature. We validate this by specifying a number of protocols in this format and analysing them with our implementation (see Section 5).

Together with the Start, Intruder and Network rules from Figure 4, the restricted format implies well-formedness of the specified protocol. We will briefly argue the validity of the properties under this restricted format. Time consistency follows from the precondition $t \geq maxt(\alpha)$ in the Intruder and Network rules and in the restricted protocol rule. Speed-of-light consistency follows from the precondition $t \geq t' + d\,(A, B)\,/\mathsf{c}$ in the Network rule and the requirement that $e \notin \mathsf{Recv}$ in the restricted protocol rule. *Prefix-closure* follows from the precondition $\alpha \in \mathsf{Tr}\,(\mathcal{P})$ in all rules, together with the fact that the conclusion extends this trace with a single event. *Time-unawareness* follows from the fact that in the construction of the traces any time $t \geq maxt(\alpha)$ is allowed for the next event, as long as *speed-of-light consistency* is satisfied. The property *locally-enabled events* follows from the requirement that a rule only concerns a single actor. The *transmission-enabled events* property follows directly from the Network rule. *Substitution-closure* expresses the (implicit) universal quantification over agents' names in all rules.

5 Causality-Based Verification

Given the definitions and properties from the previous sections, we can now formulate the notion of *causality-based secure distance-bounding* and prove that it is equivalent to the original definition of *secure distance-bounding* from Definition 1. The main feature of this new formulation is that it is causality-based, i.e., it only takes into account the relative occurrence of events, while ignoring the actual timestamps of the events and agents' locations.

This new formulation strongly relates to authentication properties, such as *aliveness* (see [18]). It states that for every claim that prover P is in the vicinity of verifier V, due to a challenge event u and the reception of its corresponding response event v in the fast phase, agent P (or a conspiring agent, if P is dishonest) must have been active in between these two events. The main difference with Definition 1 is that we require the prover to be active, instead of measuring the time between u and v.

Definition 2 (Causality-based secure distance-bounding). *A well-formed protocol $\mathcal{P}$ satisfies* causality-based secure distance-bounding *if and only if:*

$$\begin{aligned}&\forall \sigma \in \pi(\mathsf{Tr}\,(\mathcal{P})), V, P \in \mathsf{Agent}, u, v \in \mathsf{Ev}\colon \\ &\quad \mathsf{claim}_V\,(P, u, v) \in \sigma \implies \exists i, j, k \in \{1, \ldots, |\sigma|\}\colon \\ &\qquad i < j < k \wedge u = \sigma_i \wedge v = \sigma_k \wedge P \approx actor\,(\sigma_j)\,. \end{aligned} \tag{2}$$

In Definition 2 we formalize our causality-based notion of secure distance-bounding. This formulation impacts only the security analysis in the design stage. It does not affect the runtime behavior of the agents executing the protocol. In particular, the verifying agent still has to measure the round-trip time of the message exchanges in the fast phase.

In the remainder of this section, we develop the proof that the causality-based definition is equivalent to the secure distance-bounding property from Definition 1. To do so, we first present a few lemmas that follow from the basic properties of the semantic domain described in Section 4.1. They will prove useful when deriving our main result.

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

In Lemma 1 below, we demonstrate that for any well-formed protocol $\mathcal{P}$, any valid timed-trace $\alpha\cdot(t,e) \in \mathsf{Tr}\left(\mathcal{P}\right)$ must contain a subsequence β that is also a valid trace in $\mathcal{P}$, and contains (t,e) and $\psi\left(\beta\right)$. We use $|.|$ to denote the length of a (timed or not) trace, in terms of the number of events.

Lemma 1. *Let $\mathcal{P}$ be a well-formed protocol. Then the following holds:*

$$\begin{aligned}\forall\alpha \in \mathsf{Tr}\left(\mathcal{P}\right),(t,e)\in\mathbb{R}\times\mathsf{Ev}\colon \alpha\cdot(t,e)\in\mathsf{Tr}\left(\mathcal{P}\right) \implies&\\ \exists\beta\in\mathsf{Tr}\left(\mathcal{P}\right): (t,e)\in\beta\wedge\beta\sqsubseteq\alpha\cdot(t,e)\wedge\psi\left(\beta\right).&\end{aligned}$$

Proof. We will proceed by induction over $|\alpha|$. The base case $|\alpha| = 0$ trivially holds by setting $\beta = (t,e)$. So, let $n \in \mathbb{N}\setminus\{0\}$ and assume by the induction hypothesis that the lemma holds for all $\alpha\in\mathsf{Tr}\left(\mathcal{P}\right)$ with $|\alpha| < n$. Now, let $\alpha = (t_1,e_1)\cdots(t_n,e_n)\in\mathsf{Tr}\left(\mathcal{P}\right)$ and $(t,e)\in\mathbb{R}\times\mathsf{Ev}$ such that $\gamma = \alpha\cdot(t,e)\in\mathsf{Tr}\left(\mathcal{P}\right)$. Let us analyse the two cases:

Case 1 ($e \in \mathsf{Recv}$)**.** From Property 2 we have that there exists $i\in\{1,\ldots,n\}$ such that $e_i \rightsquigarrow e$ and $t - t_i \geq d\left(e_i,e\right)/\mathsf{c}$. Consider $\alpha' = (t_1,e_1)\cdots(t_{i-1},e_{i-1})$. Then, from the induction hypothesis (given that $|\alpha'| = i-1 < n$ and $\alpha'\in\mathsf{Tr}\left(\mathcal{P}\right)$ due to Properties 3 and 4) it follows that there exists $\beta'\in\mathsf{Tr}\left(\mathcal{P}\right)$ with $(t_i,e_i)\in\beta'$ such that $\beta'\sqsubseteq\alpha'$ and $\psi\left(\beta'\right)$. Thus, $\psi\left(\beta'\right)$ along with $t-t_i\geq d\left(e_i,e\right)/\mathsf{c}$ give us that $\psi\left(\beta'\cdot(t,e)\right)$ and $\beta'\cdot(t,e)$ is time and speed-of-light consistent.

Now, from Property 6 we derive $\pi(\beta')\cdot e\in\pi(\mathsf{Tr}\left(\mathcal{P}\right))$. On the other hand, $\beta'\cdot(t,e)\sim\beta''$ for some $\beta''\in\mathsf{Tr}\left(\mathcal{P}\right)$ such that $\pi(\beta')\cdot e = \pi(\beta'')$. Finally, Property 4 gives us $\beta'\cdot(t,e)\in\mathsf{Tr}\left(\mathcal{P}\right)$.

Case 2 ($e \notin \mathsf{Recv}$)**.** Let i be the largest number in $\{1,\ldots,n\}$ such that $\mathit{actor}\left(e_i\right) = \mathit{actor}\left(e\right)$. If i does not exist, then from Property 5 we obtain that $e\in\pi(\mathsf{Tr}\left(\mathcal{P}\right))$ and therefore $(t',e)\in\mathsf{Tr}\left(\mathcal{P}\right)$ for some $t'\in\mathbb{R}$. Hence, as (t,e) is time and speed-of-light consistent, Property 4 gives us $(t,e)\in\mathsf{Tr}\left(\mathcal{P}\right)$ as $(t,e)\sim(t',e)$. Further, $\psi\left((t,e)\right)$ trivially holds, which leaves us with the remaining case in which i exists.

Let $\alpha' = (t_1, e_1) \cdots (t_i, e_i)$. Then, from the induction hypothesis (given that $|\alpha'| = i - 1 < n$ and $\alpha' \in \mathsf{Tr}(\mathcal{P})$ due to Properties 3 and 4) it follows that there exists $\beta' \in \mathsf{Tr}(\mathcal{P})$ with $(t_i, e_i) \in \beta'$ such that $\beta' \sqsubseteq \alpha'$ and $\psi(\beta')$. Thus, $\psi(\beta')$ along with $t - t_i \geq d(e_i, e)/\mathsf{c} = 0$ give us that $\psi(\beta' \cdot (t, e))$ and $\beta' \cdot (t, e)$ is time and speed-of-light consistent.

Now, from Property 6 we derive $\pi(\beta') \cdot e \in \pi(\mathsf{Tr}(\mathcal{P}))$. On the other hand, $\beta' \cdot (t, e) \sim \beta''$ for some $\beta'' \in \mathsf{Tr}(\mathcal{P})$ such that $\pi(\beta'') = \pi(\beta') \cdot e$. Finally, Property 4 gives us $\beta' \cdot (t, e) \in \mathsf{Tr}(\mathcal{P})$. □

Lemma 2 below is an extension of Lemma 1. It states that if a valid trace α satisfies $\psi(\alpha)$, then not only any pair of consecutive events in α satisfy the time/location constraint but also any pair of events in α. The proof follows from the application of the triangle inequality $d(e, e')/\mathsf{c} + d(e', e'')/\mathsf{c} \geq d(e, e'')/\mathsf{c}$, for all $e, e', e'' \in \mathsf{Ev}$, given that d models physical distances.

Lemma 2. *Let $\mathcal{P}$ be a well-formed protocol and $\alpha \in \mathsf{Tr}(\mathcal{P})$ such that $\psi(\alpha)$. Then for all $(t, e), (t', e') \in \alpha$ it holds that $|t - t'| \geq d(e, e')/\mathsf{c}$.*

Proof. Let $\alpha = (t_1, e_1) \cdots (t_n, e_n)$ and $i, j \in \{1, \ldots, n\}$. Assume without loss of generality that $i < j$. Given that $\psi(\alpha)$ we have that $t_x - t_{x-1} \geq d(e_{x-1}, e_x)/\mathsf{c}$ for all $x \in \{i+1, \ldots, j\}$. Hence,

$$\begin{aligned} t_j - t_i = &\ (t_j - t_{j-1}) + (t_{j-1} - t_{j-2}) + \cdots + (t_{i+1} - t_i) \\ \geq &\ d(e_i, e_{i+1})/\mathsf{c} + d(e_{i+1}, e_{i+2})/\mathsf{c} + \cdots + d(e_{j-1}, e_j)/\mathsf{c}. \end{aligned} \tag{3}$$

Thus, by applying the triangle inequality in Equation 3 above, we obtain $t_j - t_i \geq d(e_i, e_j)/\mathsf{c}$. □

The last lemma of this section concerns agent substitutions. We extend Property 7 from the set of untimed-traces $\pi(\mathsf{Tr}(\mathcal{P}))$ of a given protocol $\mathcal{P}$ to the set of timed-traces $\mathsf{Tr}(\mathcal{P})$. The lemma proves that, given a protocol's valid trace $\alpha = (t_1, e_1) \cdots (t_n, e_n)$, it is possible to replace an agent A by another agent B (under certain conditions described in the lemma) to obtain another valid trace $\alpha' = (t'_1, e'_1) \cdots (t'_n, e'_n)$ such that the difference between t'_i and t_i only depends on the number of events before the i-th event on α that were executed by A. Consequently, the time-difference between two events of α where A does not act is equal to the time-difference between the corresponding events of α'. This is actually a strong result because it implicitly shows that event-intervals where the prover does not act cannot be used to securely upper-bound the prover-to-verifier distance.

Lemma 3. *Let $\mathcal{P}$ be a well-formed protocol and $\alpha = (t_1, e_1) \cdots (t_n, e_n) \in \mathsf{Tr}(\mathcal{P})$. Let $A \in actor(\alpha)$, $B \in \mathsf{Agent} \setminus actor(\alpha)$ such that either $\{A, B\} \subseteq \mathsf{Honest}$ or $\{A, B\} \subseteq \mathsf{Dishonest}$. Then there exists $\mu \in \mathbb{R}_{\geq 0}$ such that $\alpha' = (t'_1, e'_1) \cdots (t'_n, e'_n) \in \mathsf{Tr}(\mathcal{P})$ where for all $i \in \{1, \ldots, n\}$ it holds that:*

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

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

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

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

1. $A = \mathit{actor}(e_i)$. In this case $q_j \geq q_i + 1$ because $e_i \notin \mathsf{Recv}$. Therefore $t'_j - t'_i \geq t_j - t_i + \mu \geq d\left(e'_i, e'_j\right)/\mathsf{c}$ as $\mu \geq d\left(e'_i, e'_j\right)/\mathsf{c}$.
2. $A \neq \mathit{actor}(e_i)$ and $A = \mathit{actor}(e_j)$. In this case we have again $q_j \geq q_i + 1$ as $e_j \in \mathsf{Recv}$, and it follows analogously to the previous case.
3. $A \notin \{\mathit{actor}(e_i), \mathit{actor}(e_j)\}$. This case gives us $\mathit{actor}(e_i) = \mathit{actor}(e'_i)$ and $\mathit{actor}(e_j) = \mathit{actor}(e_j)$. Thus, $d(e_i, e_j)/\mathsf{c} = d\left(e'_i, e'_j\right)/\mathsf{c}$ and therefore $t'_j - t'_i = t_j - t_i + \mu \cdot (q_j - q_i) \geq t_j - t_i \geq d(e_i, e_j)/\mathsf{c} = d\left(e'_i, e'_j\right)/\mathsf{c}$.

Thus, α' is time consistent and speed-of-light consistent. Consider now $\sigma = \pi(\alpha)$. From Property 7 we have that $\sigma[A \mapsto B] \in \pi(\mathsf{Tr}(\mathcal{P}))$. Therefore, there exists $\gamma \in \mathsf{Tr}(\mathcal{P})$ such that $\pi(\gamma) = \sigma[A \mapsto B]$. Finally, given that $\gamma \sim \alpha'$, from Property 4 we obtain $\alpha' \in \mathsf{Tr}(\mathcal{P})$. □

Theorem 1. *A well-formed protocol $\mathcal{P}$ satisfies* secure distance-bounding *(Definition 1) if and only if $\mathcal{P}$ satisfies* causality-based secure distance-bounding *(Definition 2).*

Proof. We will proceed by proving *Sufficiency* (i.e., Equation 1 ⇒ Equation 2) and *Necessity* (i.e., Equation 2 ⇒ Equation 1):

Sufficiency. Assume Equation 1 holds and Equation 2 does not. Our goal is to reach a contradiction. The statement that Equation 2 does not hold is equivalent to stating that there exist $\sigma = \sigma_1 \cdots \sigma_n \in \pi(\mathsf{Tr}(\mathcal{P}))$, $V, P \in \mathsf{Agent}$, $u, v \in \mathsf{Ev}$ and $l \in \{1, \ldots, n\}$ such that $\sigma_l = \mathsf{claim}_V(P, u, v)$ and:

$$\begin{aligned} &\forall i, j, k \in \{1, \ldots, n\}: \\ &\quad u = \sigma_i \wedge v = \sigma_k \wedge i < j < k \implies P \not\approx \mathit{actor}(\sigma_j). \end{aligned} \tag{4}$$

Consider now the following sets:

$$\begin{aligned} \mathit{IK} &= \{(i,k) \in \mathbb{N} \times \mathbb{N} \mid \sigma_i = u \wedge \sigma_k = v\}, \\ J &= \{j \in \mathbb{N} \mid \exists (i,k) \in \mathit{IK} : i < j < k\}, \\ \{G_1, \ldots, G_g\} &= \{G \in \mathit{actor}(\sigma) \mid P \approx G\}. \end{aligned}$$

If P is honest, then the set $\{G_1, \ldots, G_g\}$ consists of the singleton $\{P\}$, otherwise it contains all dishonest agents acting in σ.

Let $Eve, Charlie \in \mathsf{Agent} \backslash actor\,(\sigma)$ be two different agents such that $\{P, Eve, Charlie\} \subseteq \mathsf{Honest}$ or $\{P, Eve, Charlie\} \subseteq \mathsf{Dishonest}$.

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

$$actor\,(e_1 \cdots e_n) = \{V, Eve\} \cup N, \text{ where } \forall E \in N\colon Eve \not\approx E. \tag{5}$$

Let $t_1, \ldots, t_n \in \mathbb{R}$ such that $(t_1, e_1) \cdots (t_n, e_n) \in \mathsf{Tr}\,(\mathcal{P})$. Observe that the t_i's exist because $e_1 \cdots e_n \in \pi(\mathsf{Tr}\,(\mathcal{P}))$. Hence, from Equations 1 and 5 and given that $e_l = \mathsf{claim}_V\,(Eve, e_i, e_k)$ for some $(i,k) \in IK$, we derive that $\delta \in \mathbb{R}_{\geq 0}$ exists such that:

$$d(V, Eve) + \delta = \frac{\mathsf{c}}{2} \max_{(i,k) \in IK} \{t_k - t_i\}. \tag{6}$$

From Lemma 3 we have that there exist $\mu \in \mathbb{R}_{\geq 0}$, $(t'_1, e'_1) \cdots (t'_n, e'_n) \in \mathsf{Tr}\,(\mathcal{P})$ and $q_1, \ldots, q_n \in \mathbb{N}$ such that for all $i \in \{1, \ldots, n\}$, $e'_i = e_i[Eve \mapsto Charlie]$ and $t'_i = t_i + \mu \cdot q_i$ (see the construction of the q_i's in Lemma 3). On the other hand, from Equation 4 we have that $\forall j \in J\colon Eve \neq actor\,(e_j)$. Therefore

$$\forall (i,k) \in IK\colon t'_k - t'_i = t_k - t_i. \tag{7}$$

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

$$actor\,\left(e'_1 \cdots e'_n\right) = \{V, Charlie\} \cup N, \text{ where } \forall C \in N\colon Charlie \not\approx C. \tag{8}$$

Again, $e'_l = \mathsf{claim}_V\,(Charlie, e'_i, e'_k)$ for some $(i,k) \in IK$, so from Equations 1 and 8 we derive:

$$d(V, Charlie) \leq \frac{\mathsf{c}}{2} \max_{(i,k) \in IK} \{t'_k - t'_i\}. \tag{9}$$

Finally, from Equations 6, 7 and 9 we derive that $d(V, Charlie) \leq d(V, Eve) + \delta$. This is a contradiction, as δ does not depend on $Charlie$ who is an *arbitrary* agent from the same set as P in Honest or $\mathsf{Dishonest}$. Therefore we can always find $Charlie$ such that his distance to V is larger than $d(V, Eve) + \delta$.

Necessity. Assume Equation 2 holds. We will prove that Equation 1 holds as well. Let $\sigma \in \pi(\mathsf{Tr}\,(\mathcal{P}))$ and $\alpha \in \mathsf{Tr}\,(\mathcal{P})$ such that $\sigma = \pi(\alpha)$. Let $V, P \in \mathsf{Agent}$, $u, v, w \in \mathsf{Ev}$ and $tw \in \mathbb{R}$ such that $(tw, w) \in \alpha$ and $w = \mathsf{claim}_V\,(P, u, v)$. Also, let $\beta \in \mathsf{Tr}\,(\mathcal{P})$ such that $\beta \sqsubseteq \alpha$, $(tw, w) \in \beta$ and $\psi\,(\beta)$. Observe that β exists because of Lemma 1.

From Equation 2 and given that $\pi(\beta) \in \pi(\mathsf{Tr}\,(\mathcal{P}))$, we have that there exist $tu', tv' \in \mathbb{R}$, $P' \in \mathsf{Agent}$ and $(t, e) \in \beta$ such that $P' = actor\,(e)$, $tu' \leq t \leq tv'$, $(tu', u) \in \beta$, $(tv', v) \in \beta$ and $P \approx P'$. Hence, Lemma 2 gives us:

$$tv' - tu' = (tv' - t) + (t - tu') \geq \frac{d(e, v) + d(u, e)}{\mathsf{c}} = 2d\left(V, P'\right)/\mathsf{c},$$

which proves Equation 1 as $(tu', u) \in \beta \sqsubseteq \alpha$, $(tv', v) \in \beta \sqsubseteq \alpha$ and $(tw, w) \in \beta \sqsubseteq \alpha$. $\square$

The result obtained from Theorem 1 means that, within the semantic domain described in Section 4.1, the secure distance-bounding property can be verified by simply analysing the ordering of events in the traces. Therefore, the notions of time and location are indeed unnecessary for the symbolic verification of distance-bounding protocols.

6 Automated Verification

We implemented the causality-based definition of secure distance-bounding in the software tool Tamarin [15]. This allowed us to automatically verify the (in)security of 13 distance-bounding protocols and their variations. The source code of our implementation can be freely accessed online[1]. A discussion on the Tamarin tool and its specifics can be found in Appendix A.

To explain the overall methodology we use to analyse distance-bounding protocols, we perform a comprehensive analysis of the Terrorist-fraud Resistant and Extractor-free Anonymous Distance-bounding (TREAD) protocol [24] in Section 6.1. Later on, in Section 6.2 we show and discuss the results of our automated verification.

6.1 Breaking the TREAD Protocol

The TREAD protocol was claimed to satisfy various security properties, making use of the computational model DFKO introduced in [8]. Relaying on this model, a proof is given to show probabilistic resistance[2] against mafia-fraud, distance-fraud, terrorist-fraud, and distance-hijacking attacks. However, by using our framework, we have identified mafia-fraud and distance-hijacking attacks on this protocol.

TREAD consists of three phases (see Figure 8). First, the prover P generates two nonces α and β, and creates the message $\sigma = \alpha|\beta|\mathsf{idpriv}(P)$, where $\mathsf{idpriv}(P)$ is an anonymous group identity. This message is signed by P and sent encrypted to the verifier V, together with P's identity $\mathsf{idpub}(P)$. Upon reception, V decrypts the message and verifies the signature. If correct, V finishes the first phase by sending a random nonce m of size n to P. The second phase is a standard n-round fast phase wherein V sends a random bit c_i with $i \in \{0, \dots, n-1\}$ and P replies back with α_i if $c_i = 0$, with $\beta_i \oplus m_i$ otherwise. The protocol finishes successfully if all responses during the fast phase are correct and the round-trip times are below a predefined threshold (third phase).

To symbolically verify TREAD, we transform the fast phase into a single challenge-response message exchange (see Figure 9). We also ignore details that are irrelevant to our security analysis, such as the anonymous identity of the prover, and upgrade bitwise operations to stronger cryptographic primitives, such as a hash function. Overall, our goal is to obtain an abstraction of the original protocol such that every attack found in the abstraction can be mapped back onto the original protocol.

TREAD can be instantiated with either a symmetric or an asymmetric encryption scheme. We thus specified in Tamarin two variants of the TREAD protocol: one where k is a symmetric key and another one where k is an asymmetric key. In the second variant, Tamarin

[1]At `http://satoss.uni.lu/software/DBVerify/`

[2]No attack succeeds with non-negligible probability.

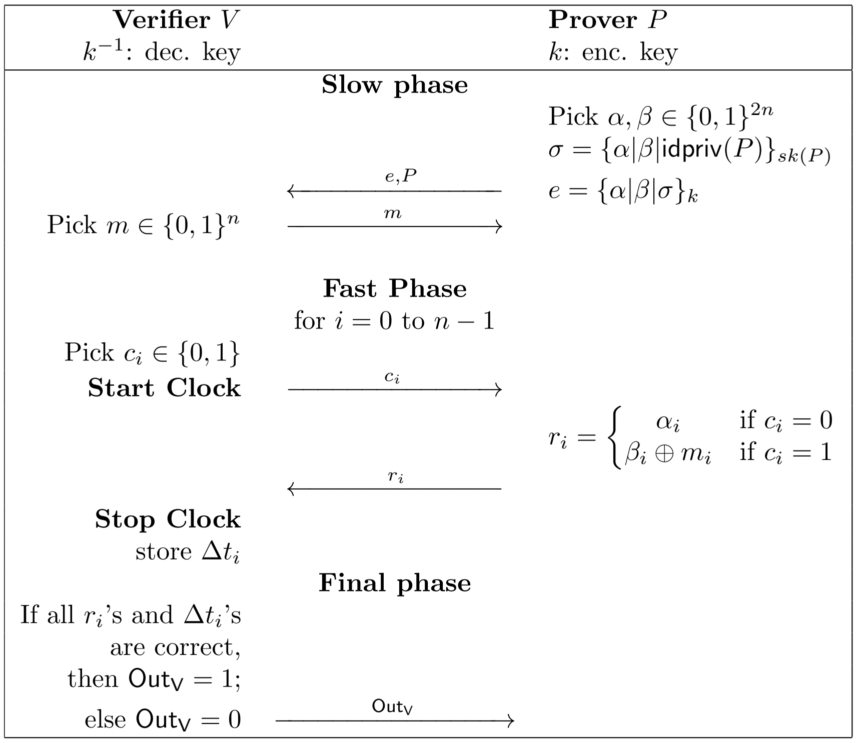

Verifier V
k^{-1}: dec. key
Prover P
k: enc. key
Slow phase
Pick $\alpha, \beta \in \{0,1\}^{2n}$
$\sigma = \{\alpha|\beta|\mathsf{idpriv}(P)\}_{sk(P)}$
e,P
$e = \{\alpha|\beta|\sigma\}_k$
Pick $m \in \{0,1\}^n$
m
Fast Phase
for $i = 0$ to $n-1$
Pick $c_i \in \{0,1\}$
Start Clock
c_i
$r_i = \begin{cases} \alpha_i & \text{if } c_i = 0 \\ \beta_i \oplus m_i & \text{if } c_i = 1 \end{cases}$
r_i
Stop Clock
store Δt_i
Final phase
If all r_i's and Δt_i's are correct,
then $\mathsf{Out_V} = 1$;
else $\mathsf{Out_V} = 0$
$\mathsf{Out_V}$

Figure 8: The TREAD protocol.

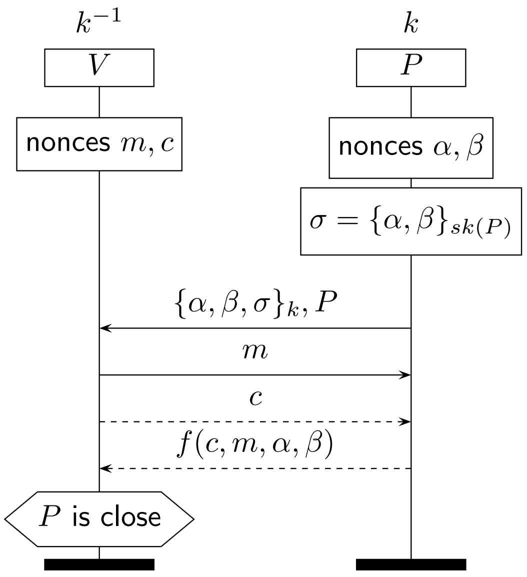


Figure 9: A representation of the TREAD protocol.

finds a simple man-in-the-middle attack that violates the secure distance-bounding property. The attack is depicted in Figure 10 and works as follows. An intruder I initiates a session with the prover P by requesting P to prove proximity. P then sends the message $(\{\alpha, \beta, \{\alpha, \beta\}_{sk(P)}\}_{pk(I)}, P)$ to I. Now the intruder decrypts the received message, learns the nonces α and β, and re-encrypts the message with the public key of the verifier. Next, the intruder starts a session with a legitimate verifier V with goal of impersonating P. To do so, I sends $(\{\alpha, \beta, \{\alpha, \beta\}_{sk(P)}\}_{pk(V)}, P)$ to V. Then V checks that the signed message $\{\alpha, \beta\}_{sk(P)}$ indeed corresponds to P, and sends back two nonces m and c. The attack ends with the intruder correctly replying to the challenges with $f(c, m, \alpha, \beta)$.

Observe that the attack described above and depicted in Figure 10 not only breaks standard authentication properties such as agreement and synchronization [39, 18], but also the secure distance-bounding property as follows. Assume P is far from V and the intruder wants to convince V that P is close. To do so, the intruder just needs to be close to V and executes the attack above. Note that the fast phase corresponds to the events containing the messages c and $f(c, m, \alpha, \beta)$, which the intruder can successfully produce without relaying.

Interesting enough, if k is a symmetric key the described mafia-fraud attack does not work. The reason is that the intruder does not know the secret key shared between P and V. Thus the intruder is prevented from re-encrypting the message received from P with the correct key. Nevertheless, a distance-hijacking type of attack exists irrespective of the encryption scheme. The attack is represented in Figure 11. Assume an honest prover P is close to the verifier V, while the intruder I is far from V. As before, P executes the protocol to prove its proximity to I. This allows I to learn α and β. Thus I starts a session with V by using the nonces α and β from P. At this point, V believes I is a legitimate prover and accept its signature. During the fast phase, P, which is close to V, receives the challenge (supposedly from I) sent by V and replies correctly. Then V receives the response $f(c, m, \alpha, \beta)$ (supposedly from I) from P who is close to V, and finishes the protocol with I correctly.

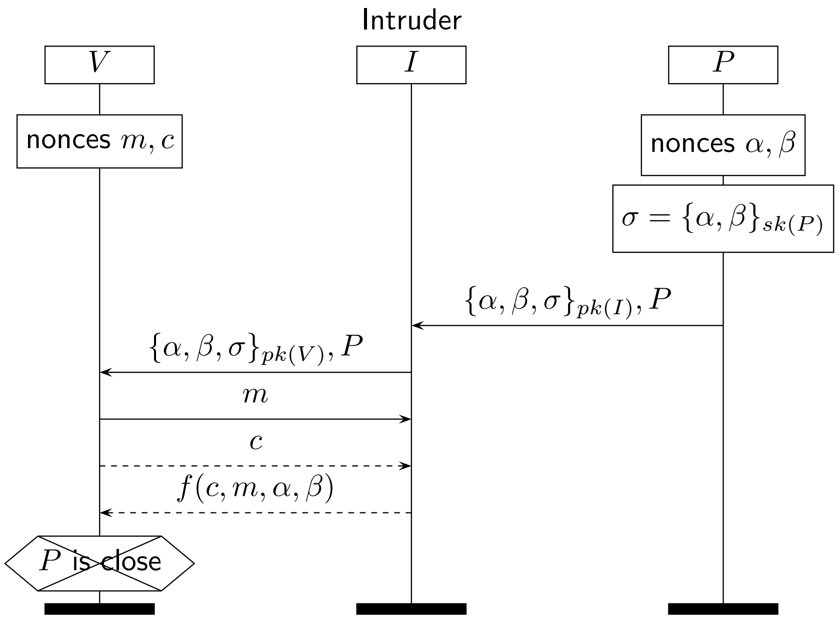
Neither of the two described attacks are possible when considering the adversary model used by the authors of the TREAD protocol, because their model does not allow for "malicious" verifiers. In their model an honest prover will fail to initiate a communication with an untrusted verifier as the first message in each attack will not be sent. This adversary model is weaker than other models that are more common in the distance-bounding literature.

6.2 Verification Results and Discussion

We applied the above analysis methodology on a number of distance-bounding protocols. To the best of our knowledge, this is the first large-scale automated security analysis of distance-bounding protocols in literature.

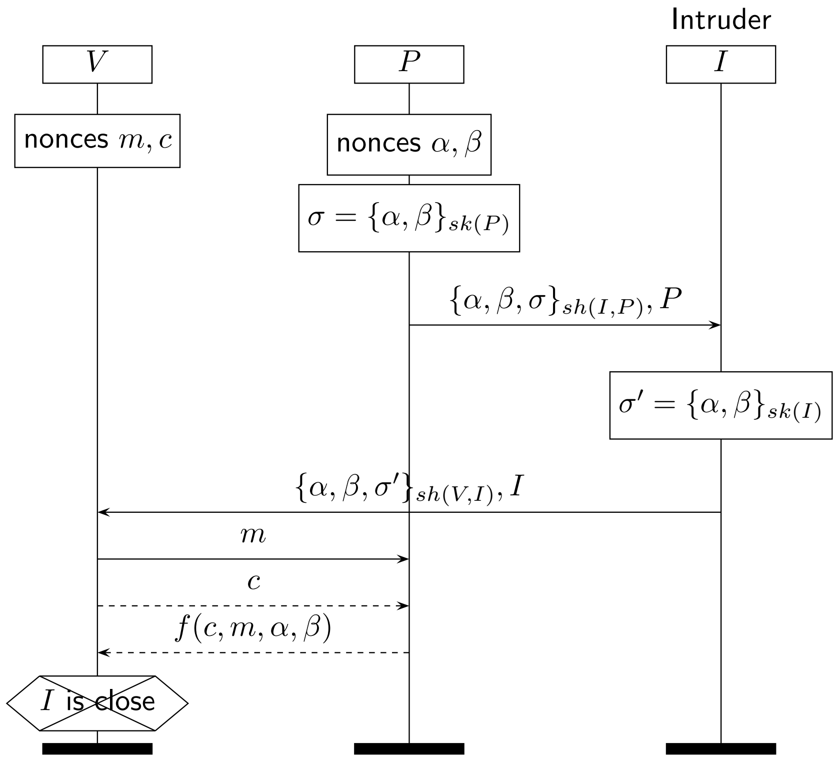
Table 1 summarizes the results of the verification. The columns *Code* and *Time* refer to the code complexity (number of lines of code) and the verification execution time (in seconds), respectively. To measure execution time, we ran the verification 10 times for each protocol and computed the average time. The column *Attacks* indicates the type of attack found (if any) by Tamarin: mafia fraud (MF), distance fraud (DF), or distance hijacking (DH). The protocols were verified by using a 64-bit Ubuntu 16.04 LTS computer with 15.5 Gb of RAM memory and a processor Intel Core i7-6700HQ CPU @ 2.60GHz $\times$ 8.

We remark that the Tree-based, Poulidor, Hancke and Kuhn's and Uniform protocols have



Intruder
V
I
P
nonces m, c
nonces α, β
$\sigma = \{\alpha, \beta\}_{sk(P)}$
$\{\alpha, \beta, \sigma\}_{pk(I)}, P$
$\{\alpha, \beta, \sigma\}_{pk(V)}, P$
m
c
$f(c, m, \alpha, \beta)$
P is close

Figure 10: A mafia fraud on TREAD with asymmetric encryption.



Intruder
V
P
I
nonces m, c
nonces α, β
$\sigma = \{\alpha, \beta\}_{sk(P)}$
$\{\alpha, \beta, \sigma\}_{sh(I,P)}, P$
$\sigma' = \{\alpha, \beta\}_{sk(I)}$
$\{\alpha, \beta, \sigma'\}_{sh(V,I)}, I$
m
c
$f(c, m, \alpha, \beta)$
I is close

Figure 11: A distance hijacking on TREAD with symmetric encryption.

Protocol	**Attacks**	**Code (lines)**	**Time (s)**
BC-Signature [5]	DH	185	5.98
BC-FiatShamir [5]*	DH, DF	189	6.51
BC-Schnorr [5]*	DH, DF	189	6.51
CRCS [21]	DH	182	5.56
Meadows et al. [23]	DH	226	18.59
Tree-based [26]**	None	186	2.51
Poulidor [27]**	None	186	2.51
Hancke and Kuhn [20]**	None	186	2.51
Uniform [28]**	None	186	2.51
Kim and Avoine [29]	None	184	1.76
Munilla et al. [30]	None	193	3.24
Reid et al. [40]	None	192	2.74
Swiss-Knife [41]	None	207	2.92
TREAD-PK [24]	MF	195	4.50
TREAD-SH [24]	DH	201	6.01
PaySafe [25]	DF	222	15.59

Table 1: Verification of a set of protocols in Tamarin. Protocols marked with * and ** have identical formalization, respectively.

equivalent Tamarin implementation as their symbolic formalization is the same. Similarly, the Brands and Chaum's (BC) protocol versions with Fiat-Shamir and Schnorr identification schemes have also the same representation. When verifying these two versions of the protocol, we found a distance-fraud attack against them. However, as the authors have pointed out, such an attack is no longer possible if a challenge/response causal relation is used during the fast phase, such as the XOR operation employed in the signature-based version of the protocol.

On average, the Tamarin implementation of a protocol consists of 194 lines of code and the verification takes 5.62 seconds. A total of 5 protocols (and their variations) were found vulnerable to attacks, of which 3 had been already reported flawed in the literature. The two remaining protocols are TREAD (whose analysis was detailed in Section 6.1) and PaySafe [25].

Figure 12 shows a representation of the PaySafe protocol, which is a variant of the classical EMV contactless payment protocol. PaySafe features a distance-bounding mechanism to avoid relay attacks. Although not in contradiction with the authors' claim regarding PaySafe security, our Tamarin verification found a successful distance-fraud attack against it. This attack is possible as there is no causal relation between the challenge and response in the fast phase (dashed arrows). Consequently, a dishonest prover C can send (ATC, n_C) before receiving $(UN, amount)$. A simple solution to this attack is to include the nonce UN in the response message.

When considering distance hijacking, our security analysis is consistent with the analysis performed by Cremers et al. in [10]. That is to say, in general protocols based on the Brand and Chaum's design (see Section 2, third paragraph) are vulnerable to this type of attacks, whereas those based on Hancke and Kuhn's are not. In addition, we observed that protocols

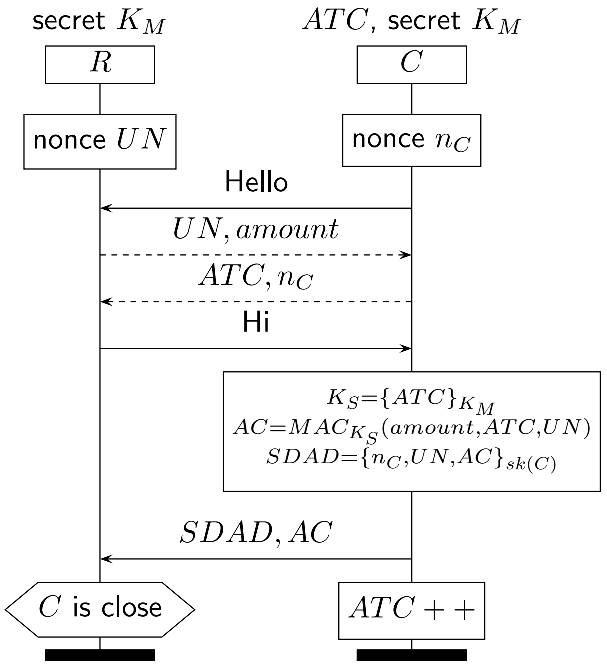


Figure 12: A representation of the PaySafe protocol.

following Hancke and Kuhn's approach seem to be resistant not only to distance hijacking but also to mafia and distance frauds.

A few of the analysed protocols have been automatically verified in previous works. Those protocols are Brands and Chaum's and its variations, as well as Meadows et al's with $F(...) = (N_V, N_P \oplus P)$, which were verified in Isabelle/HOL[3] by using Basin et al's model [19, 13]. No formal *symbolic* verification (automatic or not) has been reported for the rest of the protocols from Table 1.

Our method compares well with the Isabelle/HOL implementation of Basin et al. While our approach is fully automatic, proving a protocol insecure with Isabelle/HOL requires user-assistance to prove the existence of an attack trace. In addition, the code complexity of a protocol when implemented in Isabelle/HOL tends to be much larger. For example, the implementation of Brands and Chaum's protocol consists of 185 lines of Tamarin code, whilst the Isabelle implementation (including attack trace) takes 653.

7 Conclusion and Future Work

In this work, we addressed the topic of formal verification of distance-bounding protocols. We described and analysed a tool-supported verification framework by Basin et al. [19, 13] based on timed-events and agents' locations. By considering the language and semantics of this formalism, we characterized a semantic domain of *well-formed* distance-bounding protocols in which the timestamps associated to the agents' actions are only utilized for proximity verification purposes and not for, e.g., taking a different branch in the execution. This is not

[3]At `http://www.infsec.ethz.ch/research/software/protoveriphy.html`

a trivial class of distance-bounding protocols but it contains, to the best of our knowledge, all protocols published to date. Our main result consists of the first causality-based security model for symbolic verification of distance-bounding protocols, which we prove equivalent to Basin et al's model.

Our proposal does not consider time and location, but is instead based on the order of events in the execution traces. By implementing our proposed model in the Tamarin verification tool, we automatically verified various state-of-the-art protocols. It is therefore the first fully automated formal verification framework for distance-bounding protocols. With our automated verification, we identified unreported vulnerabilities in two recent protocols: a mafia-fraud and a distance-hijacking attack on the TREAD protocol [24], and a distance-fraud attack against the EMV-based contactless payment protocol PaySafe [25].

As future work, we plan to extend the formalism to capture terrorist-fraud attacks (which are not covered in Basin et al's model either), and formalize the different attacks and protocols' characteristics to prevent them. We also plan to extend our methodology as to capture probabilistic reasoning in a causality-based model. This will allow us to automatically determine the probability of success of a given attack against a distance-bounding protocol.

References

[1] Y. Desmedt, C. Goutier, and S. Bengio, "Special uses and abuses of the fiat-shamir passport protocol," in *CRYPTO'87*, 1987, pp. 21–39.

[2] A. Francillon, B. Danev, and S. Capkun, "Relay attacks on passive keyless entry and start systems in modern cars," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*, 2011.

[3] S. Bengio, G. Brassard, Y. Desmedt, C. Goutier, and J. Quisquater, "Secure implementations of identification systems," *J. Cryptology*, vol. 4, no. 3, pp. 175–183, 1991.

[4] T. Beth and Y. Desmedt, "Identification tokens - or: Solving the chess grandmaster problem," in *CRYPTO'90*, 1990, pp. 169–177.

[5] S. Brands and D. Chaum, "Distance-bounding protocols," in *EUROCRYPT'93*, 1993, pp. 344–359.

[6] Y. Desmedt, "Major security problems with the 'unforgeable'(feige)-fiat-shamir proofs of identity and how to overcome them," in *SECURICOM'88*, 1988, pp. 15–17.

[7] G. Avoine, M. A. Bingöl, S. Kardas, C. Lauradoux, and B. Martin, "A framework for analyzing RFID distance bounding protocols," *Journal of Computer Security*, vol. 19, no. 2, pp. 289–317, 2011.

[8] U. Dürholz, M. Fischlin, M. Kasper, and C. Onete, "A formal approach to distance-bounding RFID protocols," in *Information Security, 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings*, 2011, pp. 47–62.

[9] I. Boureanu and S. Vaudenay, “Optimal proximity proofs,” in *Information Security and Cryptology - 10th International Conference, Inscrypt 2014, Beijing, China, December 13-15, 2014, Revised Selected Papers*, 2014, pp. 170–190.

[10] C. J. F. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun, “Distance hijacking attacks on distance bounding protocols,” in *S&P'12*, 2012, pp. 113–127.

[11] D. Dolev and A. C. Yao, “On the security of public key protocols,” *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–207, 1983.

[12] I. C. Boureanu, A. Mitrokotsa, and S. Vaudenay, “Towards secure distance bounding,” in *Fast Software Encryption – 20th International Workshop, FSE 2013*, ser. LNCS, S. Moriai, Ed., vol. 8424. Singapore, Republic of Singapore: Springer, February 2013, invited Talk by Serge Vaudenay.

[13] P. Schaller, B. Schmidt, D. A. Basin, and S. Capkun, “Modeling and verifying physical properties of security protocols for wireless networks,” in *CSF'09*, 2009, pp. 109–123.

[14] T. Nipkow, L. C. Paulson, and M. Wenzel, *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*, ser. LNCS. Springer, 2002, vol. 2283.

[15] S. Meier, B. Schmidt, C. Cremers, and D. A. Basin, “The TAMARIN prover for the symbolic analysis of security protocols,” in *CAV'13*, 2013, pp. 696–701.

[16] B. Blanchet, “An Efficient Cryptographic Protocol Verifier Based on Prolog Rules,” in *CSFW'01*, 2001, pp. 82–96.

[17] C. J. F. Cremers, “The Scyther tool: Verification, falsification, and analysis of security protocols,” in *CAV'08*, 2008, pp. 414–418.

[18] C. Cremers and S. Mauw, *Operational Semantics and Verification of Security Protocols*. Springer, 2012.

[19] D. A. Basin, S. Capkun, P. Schaller, and B. Schmidt, “Let's get physical: Models and methods for real-world security protocols,” in *TPHOLs'09*, 2009, pp. 1–22.

[20] G. P. Hancke and M. G. Kuhn, “An RFID distance bounding protocol,” in *SecureComm'05*, 2005, pp. 67–73.

[21] K. B. Rasmussen and S. Capkun, “Realization of RF distance bounding,” in *USENIX Security'10*, 2010, pp. 389–402.

[22] S. Capkun, L. Buttyán, and J. Hubaux, “SECTOR: secure tracking of node encounters in multi-hop wireless networks,” in *Proceedings of the 1st ACM Workshop on Security of ad hoc and Sensor Networks, SASN 2003, Fairfax, Virginia, USA, 2003*, 2003, pp. 21–32.

[23] C. A. Meadows, R. Poovendran, D. Pavlovic, L. Chang, and P. F. Syverson, “Distance bounding protocols: Authentication logic analysis and collusion attacks,” in *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*, 2007, pp. 279–298.

[24] G. Avoine, X. Bultel, S. Gambs, D. Gérault, P. Lafourcade, C. Onete, and J. Robert, "A terrorist-fraud resistant and extractor-free anonymous distance-bounding protocol," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, 2017, pp. 800–814.

[25] T. Chothia, F. D. Garcia, J. de Ruiter, J. van den Breekel, and M. Thompson, "Relay cost bounding for contactless EMV payments," in *Financial Cryptography and Data Security - 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers*, 2015, pp. 189–206.

[26] G. Avoine and A. Tchamkerten, "An efficient distance bounding RFID authentication protocol: Balancing false-acceptance rate and memory requirement," in *ISC'09*, 2009, pp. 250–261.

[27] R. Trujillo-Rasua, B. Martin, and G. Avoine, "The poulidor distance-bounding protocol," in *RFIDSec'10*, 2010, pp. 239–257.

[28] S. Mauw, J. Toro-Pozo, and R. Trujillo-Rasua, "A class of precomputation-based distance-bounding protocols," in *EuroS&P'16*, 2016, pp. 97–111.

[29] C. H. Kim and G. Avoine, "RFID distance bounding protocol with mixed challenges to prevent relay attacks," in *CANS'09*, 2009, pp. 119–133.

[30] 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*, vol. 8, no. 9, pp. 1227–1232, 2008.

[31] S. Mauw, J. Toro-Pozo, and R. Trujillo-Rasua, "Optimality results on the security of lookup-based protocols," in *Radio Frequency Identification and IoT Security - 12th International Workshop, RFIDSec 2016, Hong Kong, China, November 30 - December 2, 2016, Revised Selected Papers*, 2016, pp. 137–150.

[32] A. O. Gürel, A. Arslan, and M. Akgün, "Non-uniform stepping approach to RFID distance bounding problem," in *DPM'10/SETOP'10*, ser. LNCS, vol. 6514, 2011, pp. 64–78.

[33] S. Kardas, M. S. Kiraz, M. A. Bingöl, and H. Demirci, "A novel RFID distance bounding protocol based on physically unclonable functions," in *RFIDSec'11*, ser. LNCS, vol. 7055. Springer, 2012, pp. 78–93.

[34] C. H. Kim and G. Avoine, "RFID distance bounding protocols with mixed challenges," *IEEE Trans. on Wireless Comm.*, vol. 10, no. 5, pp. 1618–1626, 2011.

[35] S. Malladi, B. Bruhadeshwar, and K. Kothapalli, "Automatic analysis of distance bounding protocols," *CoRR*, vol. abs/1003.5383, 2010.

[36] C. Pöpper, N. O. Tippenhauer, B. Danev, and S. Capkun, "Investigation of signal and message manipulations on the wireless channel," in *ESORICS'11*, 2011, pp. 40–59.

[37] M. Fischlin and C. Onete, "Terrorism in distance bounding: Modeling terrorist-fraud resistance," in *Proceedings of the 11th International Conference on Applied Cryptography and Network Security*, ser. ACNS'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 414–431.

[38] S. Vaudenay, "On modeling terrorist frauds," in *Proceedings of the 7th International Conference on Provable Security - Volume 8209*, ser. ProvSec 2013. New York, NY, USA: Springer-Verlag New York, Inc., 2013, pp. 1–20.

[39] G. Lowe, "A hierarchy of authentication specifications," in *Proceedings 10th Computer Security Foundations Workshop*, Jun 1997, pp. 31–43.

[40] J. Reid, J. M. G. Nieto, T. Tang, and B. Senadji, "Detecting relay attacks with timing-based protocols," in *Proceedings of the 2007 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2007, Singapore, March 20-22, 2007*, 2007, pp. 204–213.

[41] C. H. Kim, G. Avoine, F. Koeune, F. Standaert, and O. Pereira, "The swiss-knife RFID distance bounding protocol," in *Information Security and Cryptology - ICISC 2008, 11th International Conference, Seoul, Korea, December 3-5, 2008, Revised Selected Papers*, 2008, pp. 98–115.

[42] J. Dreier, C. Duménil, S. Kremer, and R. Sasse, "Beyond subterm-convergent equational theories in automated verification of stateful protocols," in *International Conference on Principles of Security and Trust*. Springer, 2017, pp. 117–140.

[43] M. Arapinis, J. Phillips, E. Ritter, and M. D. Ryan, "Statverif: Verification of stateful processes," *Journal of Computer Security*, vol. 22, no. 5, pp. 743–821, 2014.

A Tamarin Specifics

In this section we discuss relevant aspects of the Tamarin tool and the implementation of the secure distance-bounding property. First, a brief overview of the tool as well as its strengths and limitations are given. An introduction to the specification language that Tamarin uses follows. Finally, we provide a specification of the causality-based secure distance-bounding security claim inside Tamarin, and discuss some of the details behind how the tested protocols were implemented.

A.1 The Tamarin Prover Tool

The Tamarin prover is a software tool for automatic verification of security protocols. Tamarin makes use of multiset rewriting systems to specify the protocols.

The Tamarin implementations are in some cases not able to precisely reproduce the original protocols, and must make overestimations. Although Tamarin allows for a user-defined equational theory, it is required to be subterm convergent. Ongoing work, such as that of Dreier et al. [42] continues to reduce these restrictions.

The stateful environment that Tamarin uses means that it will never identify false attacks (e.g., it will never claim that there is an attack when one does not exist). This is in contrast to other verification tools, such as ProVerif [43], which can find false attacks. In the other direction, Tamarin is also sound in its analysis: if a claim is verified, then there indeed exist no traces that violate this claim. This property extends to the protocol that is being modelled, up to the accuracy of the implementation of protocol rules and equational theory. For example, Tamarin does not identify the probabilistic attack on Hancke and Kuhn's protocol (mentioned in Section 2), as the function used during the fast phase is abstracted to a secure hash function. Tamarin is not guaranteed to terminate for all claims, but in our case study no such problems arose, with all protocols taking less than 20 seconds (< 6 on average) to either successfully verify or provide an attack trace.

A.2 Tamarin Specification Language

Tamarin uses a specification language built on multiset rewriting. The elements of the multiset are *facts*: first order logic terms built from a fact name and a number of subterms. Subterms are built from atoms of type Pub or Fresh. Note that Fresh corresponds directly to Nonce as defined in the paper, and that Pub is assumed by design to include Agent from the paper as a subset. Msg is a superset of these, and security claims can make use of terms of type $\mathsf{Temporal}$ (a discrete type with partial order). However, atoms of type $\mathsf{Temporal}$ are *never* used inside a protocol rule. Atoms can also be either *variables* or *constants*. Constants are almost always public (and indeed are in all of our implemented protocols).

Atoms are extended to the full set of terms by function symbols and an equational theory. These are defined entirely by the user, with the only exception being the 2-ary function symbol $\langle \_, \_ \rangle$, 1-ary function symbols $\mathit{fst}(\_)$ and $\mathit{snd}(\_)$ and the equations

$$\mathit{fst}(\langle x, y \rangle) = x \qquad \mathit{snd}(\langle x, y \rangle) = y$$

In general, equational theories must be subterm convergent. Tamarin allows the importing of some stronger pre-built equational theories with more capabilities, such as fully modelling

```
lemma Secrecy:
"
All n #t. Secret(n)@t ==>
    not( Ex #s K(x)@s )
"
```

Figure 13: The Tamarin lemma Secrecy.

the exponentiation rules necessary for using the Diffie-Hellman scheme.

Tamarin reserves some fact names. Fr(n) represents the generation of a fresh nonce n. In(x) and Out(x) represent the sending or receiving of the message x from the network. K(x) represents the adversary knowing the term x. Unless the type of an atom is explicitly denoted, it is always assumed to be of type Msg. For clarity's sake, it is assumed that the arity of a fact name is constant over all instances of it in a protocol specification.

A protocol rule is written as a 'left-hand side' and a 'right-hand side', each a sequence of facts. The starting state of the protocol is the empty multiset. If the variables in the left-hand side of a rule can be instantiated in such a way that it is a subset of the current multiset, then the rule can be executed, with the (instantiated) facts in the left-hand side removed from the state and replaced with the facts in the right-hand side.

Tamarin supports two special kind of facts. *Persistent* facts may appear on the left-hand side of a rewrite rule, but are *not* removed from the multiset as a consequence of its execution. *Action* facts may only appear on the right-hand side of a protocol rule. All security claims are made based on the presence and ordering of these Action facts, and so protocol rules are in fact written as:

```
[ LHS ]--[ ACTION ]->[ RHS ]
```

For example, the sending of a freshly generated nonce onto the network might be written as:

```
[ Fr(~n) ]--[ SendNonce(~n) ]->[ Out(~n) ]
```

where $\sim$ indicates that n is of the type Fresh.

Security claims are first order logic statements based on the action facts in the protocol specification. For example, suppose that a protocol's intended execution ends with the rule:

```
[ In(n) ]--[ Secret(n) ]->[]
```

The fact name Secret(n) has been chosen to suggest that the agent enacting the rule believes the value n to be a secret. The corresponding security claim might read as in Figure 13, which is interpreted as: "For any value n and event t such that it is believed that n is secret at event t, is it not the case that there is a corresponding event at s in which the adversary knows n".

The Tamarin adversary follows the Dolev-Yao model [11], assuming full control over the network. It automatically makes use of the equational theory provided to deduce as much information as possible, and has free choice in the order of protocol rules chosen and the choice of instantiation for the variables. We provide the adversary with the additional power to corrupt certain agents, gaining knowledge of their long term secrets. Security claims are made with this in mind, in line with how the language introduced in the paper does not discriminate between dishonest agents.

A.3 Implementation Details

The definition of the *causality-based secure distance-bounding* property is not immediately compatible with the specification model that Tamarin uses. We note the following factors:

1. The definition uses a claim event that refers specifically to other events (which mark the start and end of the fast phase). Tamarin does not allow for an event to be one of the subterms of a fact.

2. The specification language partitions agents into the sets Honest and Dishonest of honest agents (who attempt to perfectly follow the protocol's intended execution) and dishonest agents (who are willing to make use of other rules in order to violate security properties). Tamarin carries no understanding of the intended execution of a protocol. Further, Tamarin does not inherently carry the notion of agents, although they are trivially modelled by public variables. The adversary is *not* considered an agent in Tamarin, and the sending of messages by the adversary is modelled using built-in rewriting rules that are often not straightforward to write claims around.

3. The security property is dependent on the identity of the *actor* of an event: i.e. the agent who performed the action. Tamarin does not explicitly attach an identity to a rewriting rule's application, as a consequence of agents not being an inherent feature of Tamarin.

These issues were addressed as follows:

1. In order to model claim events, state facts containing session data were used. In particular, the VerifierComplete(*params*) state fact was added to all rewrite rules designed to symbolise that the verifier role believes they successfully completed the protocol with session data *params*. The term *params* is built from all session data added to the protocol in the order that it is added. This necessarily includes two public variables to model the identities of the prover and verifier, as well as at least one fresh variable used in the protocol execution. However, different protocols make use of different numbers of fresh variables, so the number of them inside this state fact varies slightly. The state facts StartFastPhase(*data*) and EndFastPhase(*data*) model the start and end of the fast phase as defined by the protocol specification. Note that in protocols involving pre-commitments, the verifier is not fully aware of the value of all of the session data at the start or end of the fast phase, and so it is not necessarily the case that *params* and *data* will be equal. However, the subterms of *data* are a strict subset of the subterms of *params*.

   Assuming that the session data of a protocol is different between different runs of the protocol, the subterms of the VerifierComplete fact refer unambiguously to the corresponding state facts for the denoting fast phase. If a protocol does not have different session data between executions, it is trivially vulnerable to replay attacks.

2. Agents are modelled in protocol specifications as public terms. Rewrite rules are included to model an agent receiving any secret keys or other information they have at the start of a protocol's execution. In this case, we see facts of the form $\mathrm{Ltk}(A, k)$, denoting that the agent A has key k. Additional rewrite rules are added for the corruption of agents (in which a fact containing a secret key is sent on to the network, revealing the identity of the agent), and also to model a corrupt agent sending a message on the network. This is important for adding state facts to symbolise the adversary acting during the fast phase.

3. For the secure distance-bounding claim to make sense the identity of the prover must be used in the protocol in some way: either their identity is used in a message, or the prover possesses a long-term secret key used in a calculation. This could be for symmetric or asymmetric encryption, or in some cases for signed hashes. Any multiset rule that uses an agent's identity (or carries session data from an earlier rule in the protocol which does) is marked with the state fact $\mathrm{Action}(agent)$.

With these in mind, the Tamarin lemma `dbsec` is defined in Figure 14. This lemma can be understood as meaning that whenever a verifier reaches the end of their protocol execution, one of three following events is possible:

1. The verifier is corrupt: they have revealed their long term secret key to the adversary, making their claim invalid.

2. Between the start and end of the fast phase, the agent P that the verifier believes is close performed some action.

3. The agent P that the verifier believes is close has revealed their long term secret key to the adversary. Between the start and end of the fast phase, some corrupt agent (who may be P or another agent who has revealed their long-term key) performed an action.

Our implementations of the protocols also involve a number of *reachability* lemmas. These lemmas are not related to the main `dbsec` lemma, but instead prove that the protocol has been implemented in such a way that the various stages of the protocol can be reached as per their intended execution. If the end of the protocol is not reachable, then the `dbsec` property is trivially true.

Finally, the protocol implementations include some *trace restrictions*, also known as *axioms*. These are claims that are assumed to be true when Tamarin constructs proofs for the lemmas. The main axioms used are `at_most_once` (Figure 15) and `equality` (Figure 16).

The axiom `at_most_once` is used to ensure that a single agent (or pair of agents) may only be given a single long term key (or shared key, respectively), and the axiom `equality` serves to verify that an equation holds: typically used in the case of verifying that a signature lines up with the message it is intended to be signing.

A.4 Expressiveness and Abstraction

The Tamarin verifier supports *stateful* protocol specifications: where the output of the system is dependent on the internal state, potentially causing repeated executions to have differing consequences. The specification model provided in Section 5 is built in a stateless manner,

```
lemma dbsec:
"
All P V m n #t. (
VerifierComplete(P, V, m, n)@t ) ==>
(
  Ex #tc.
    Corrupt(V)@tc
)|(
  Ex #t1 #t2 #t3.
    StartFastPhase(V, m)@t1 &
    Action(P)@t2 &
    EndFastPhase(V, m)@t3 &
    (#t1 < #t2) &
    (#t2 < #t3) &
    ( (#t3 < #t ) | (#t3 = #t) )
)|(
  Ex CAgent #t4 #t5 #t6 #t7.
    StartFastPhase(V, m)@t5 &
    EndFastPhase(V, m)@t7 &
    Corrupted(P, V)@t4 &
    CAction(CAgent)@t6 &
    (#t5 < #t6)&
    (#t6 < #t7)&
    ( (#t7 < #t) | (#t7 = #t) )
)
"
```

Figure 14: The Tamarin lemma `dbsec`.

```
axiom at_most_once:
"
All A #t1 #t2.
Once(A)@t1 & Once(A)@t2 ==>
(
   #t1 = #t2
)
"
```

Figure 15: The Tamarin axiom `at_most_once`.

```
axiom equality:
"
All a b #t1. Eq(a, b)@t1 ==> a = b
"
```

Figure 16: The Tamarin axiom `equality`.

but allows for this extension in a natural way. Of the protocols analysed, the PaySafe protocol is the only one which contains stateful components (in the form of an incrementing counter, *ATC* in Figure 12). The attack provided on PaySafe can be identified in a stateless variant of the protocol and shown to hold even in the full version.

A key advantage with our form of automated verification is that the protocol specification and associated security claim are disjoint (save for markers indicating which events the claim is made about). This means that the security definition does not need to be adapted in order to be compatible with a given protocol, and the analyst can focus on attempting to represent the protocol's specification as faithfully as possible. There is occasional need for the analyst to make decisions about the abstraction of certain details (such as which aspects of the equational theory behind the exclusive-OR operator should be available), but this is a consequence of the limitations of the verification tool, not the specification model.