Let's Get Physical: Models and Methods for Real-World Security Protocols

David Basin, Srdjan Capkun, Patrick Schaller, and Benedikt Schmidt
ETH Zurich, 8092 Zurich, Switzerland

Abstract. Traditional security protocols are mainly concerned with key establishment and principal authentication and rely on predistributed keys and properties of cryptographic operators. In contrast, new application areas are emerging that establish and rely on properties of the physical world. Examples include protocols for secure localization, distance bounding, and device pairing.
We present a formal model that extends inductive, trace-based approaches in two directions. First, we refine the standard Dolev-Yao model to account for network topology, transmission delays, and node positions. This results in a distributed intruder with restricted, but more realistic, communication capabilities. Second, we develop an abstract message theory that formalizes protocol-independent facts about messages, which hold for all instances. When verifying protocols, we instantiate the abstract message theory, modeling the properties of the cryptographic operators under consideration. We have formalized this model in Isabelle/HOL and used it to verify distance bounding protocols where the concrete message theory includes exclusive-or.

1 Introduction

Situating Adversaries in the Physical World. There are now over three decades of research on symbolic models and associated formal methods for security protocol verification. The models developed represent messages as terms rather than bit strings, take an idealized view of cryptography, and focus on the communication of agents over a network controlled by an active intruder. The standard intruder model used, the Dolev-Yao model, captures the above aspects. Noteworthy for our work is that this model abstracts away all aspects of the physical environment, such as the location of principals and the speed of the communication medium used. This is understandable: the Dolev-Yao model was developed for authentication and key-exchange protocols whose correctness is independent of the principals' physical environment. Abstracting away these details, effectively by identifying the network with the intruder, results in a simpler model that is adequate for verifying such protocols.

With the emergence of wireless networks, protocols have been developed whose security goals and assumptions differ from those in traditional wireline networks. A prominent example is distance bounding [1–5], where one device must determine an upper bound on its *physical* distance to another, potentially untrusted, device. The goal of distance bounding is neither message secrecy nor authentication, but rather to establish a physical property. To achieve this, distance bounding protocols typically combine cryptographic guarantees, such as message-origin authentication, with properties of the physical (communication) layer, for example that attackers cannot relay messages between locations faster than the speed of light. Other examples of "physical protocols" include secure time synchronization, wormhole and neighborhood detection, secure localization, broadcast authentication, and device pairing.

In [6], we presented the first formal model that is capable of modeling and reasoning about a wide class of physical protocols and their properties. The key idea is to reflect relevant aspects of the physical world in the model, namely network topology, transmission delays, and node positions. In particular, all agents are modeled as network nodes. This includes the intruder, who is no longer a single entity but instead is distributed and therefore corresponds to a set of nodes. Communication between nodes is subject to restrictions reflecting the nodes' physical environment and communication capabilities. For example, not all nodes can communicate and communication takes time determined by the network topology and the propagation delays of the communication technologies used. Hence, nodes require time to share their knowledge and information cannot travel at speeds faster than the speed of light. Possible communication histories are formalized as traces and the resulting model is an inductively-defined, symbolic, trace-based model, along the lines of Paulson's *Inductive Approach* [7].

In [6], we formalized this model in Isabelle/HOL [8] and verified the security properties of three physical protocols: an authenticated ranging protocol [9], a protocol for distance bounding using ultrasound [5], and a broadcast-authentication protocol based on delayed key disclosure [10].

Verifying distance bounding protocols. Our starting point in this paper is a family of distance bounding protocols proposed by Meadows [4]. The family is defined by a protocol pattern containing a function variable F, where different instances of F result in different protocols. We present two security properties, which distinguish between the cases of honest and dishonest participants. For each property, we reduce the security of a protocol defined by an instance of F to conditions on F. Afterwards, we analyze several instances of F, either showing that the conditions are fulfilled or presenting counterexamples to the security properties.

This protocol family is interesting as a practically-relevant case study in applying our framework to formalize and reason about nontrivial physical protocols. Moreover, it also illustrates how we can extend our framework (originally defined over a free term algebra) to handle protocols involving equationally-defined operators on messages and how this can be done in a general way. Altogether, we have worked with five different protocols and two different message theories. To support this, we have used Isabelle's locales construct to formalize an abstract message theory and a general theory of protocols. Within the locales, we prove general, protocol-independent facts about (abstract) messages, which hold when we subsequently instantiate the locales with our different concrete message theories and protocols.

Contributions. First, we show that our framework for modeling physical security protocols can be extended to handle protocols involving equationally-defined operators. This results in a message theory extended with an *XOR* operator and a zero element, consisting of equivalence classes of messages with respect to the equational theory of *XOR*. We use normalized terms here as the representatives of the equivalence classes. With this extension, we substantially widen the scope of our approach. Note that this extension is actually independent of our "physical" refinement of communication and also could be used in protocol models based on the standard Dolev-Yao intruder.

Second, we show how such extensions can be made in a generic, modular way. Noteworthy here is that we could formulate a collection of message-independent and protocol-independent facts that hold for a large class of intended extensions. An example of such a fact is that the minimal message-transmission time between two agents A and B determines a lower bound on the time difference between A creating a fresh nonce and B learning it.

Finally, physical protocols often contain time-critical steps, which must be optimized to reduce computation and communication time. As a result, these steps typically employ low-level operations like *XOR*, in contrast to more conventional protocols where nanosecond time differences are unimportant. Our experience indicates that the use of such low-level, equationally-defined operators results in substantial additional complexity in reasoning about protocols in comparison to the standard Dolev-Yao model. Moreover, the complexity is also higher because security properties are topology dependent and so are attacks. Attacks now depend not only on what the attackers know, but also their own physical properties, i.e., the possible constellations of the distributed intruders. Due to this complexity, pencil-and-paper proofs quickly reach their limits. Our work highlights the important role that Formal Methods can play in the systematic development and analysis of physical protocols.

Organization. In Section 2, we provide background on Isabelle/HOL and the distance bounding protocols that we analyze in this paper. In Section 3, we present our formal model of physical protocols, which we apply in Section 4. Finally, in Section 5, we discuss related work and draw conclusions.

2 Background

2.1 Isabelle/HOL

Isabelle [8] is a generic theorem prover with a specialization for higher-order logic (HOL). We will avoid Isabelle-specific details in this paper as far as possible or explain them in context, as needed.

We briefly review two aspects of Isabelle/HOL that are central to our work. First, Isabelle supports the definition of (parameterized) inductively-defined sets.

An inductively-defined set is defined by sets of rules and denotes the least set closed under the rules. Given an inductive definition, Isabelle generates a rule for proof by induction.

Second, Isabelle provides a mechanism, called *locales* [11] that can be used to structure generic developments, which can later be specialized. A locale can be seen as either a general kind of proof context or, alternatively, as a kind of parameterized module. A locale declaration contains:

- a name, so that the locale can be referenced and used,
- typed parameters, e.g., ranging over relations or functions,
- assumptions about the parameters (the module axioms), and
- functions defined using the parameters.

In the context of a locale, one can make definitions and prove theorems that depend on the locale's assumptions and parameters. Finally, a locale can be interpreted by instantiating its parameters so that the assumptions are theorems. After interpretation, not only can the assumptions be used for the instance, but also all theorems proved and definitions made in the locale's context.

2.2 Distance Bounding Protocols

Distance bounding protocols are two-party protocols involving a *verifier* who must establish a bound on his distance to a *prover*. These protocols were originally introduced in [1] to prevent a man-in-the-middle attack called Mafia Fraud. Suppose, for example, that an attacker possesses a fake automated teller machine (ATM). When a user uses his banking card to authenticate himself to the fake ATM, the attacker simply forwards the authenticating information to a real ATM. After successful authentication, the attacker can plunder the user's account. Distance bounding protocols prevent this attack by determining an upper bound on the distance between the ATM and the banking card. The ATM is the verifier and checks that the card, acting as the prover, is sufficiently close by to rule out the man-in-the-middle.

The idea behind distance bounding is simple. The verifier starts by sending a challenge to the prover. The prover's reply contains an authenticated message involving the challenge, which shows that it has been received by the prover. After receiving the reply, the verifier knows that the challenge has traveled back and forth between him and the prover. Assuming that the signal encoding the challenge travels with a known speed, the verifier can compute an upper bound on the distance to the prover by multiplying the measured round-trip time of his challenge by the signal's velocity.

For distance bounding to yield accurate results, the verifier's round-trip time measurement should correspond as close as possible to the physical distance between the prover and verifier. This is achieved by having the prover generate his response as quickly as possible. Expensive cryptographic operations such as digital signatures should therefore be avoided. A distance bounding protocol can typically be decomposed into three phases: a *setup phase*, a *measurement phase*,

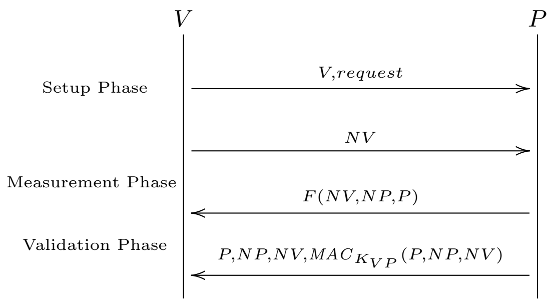

Fig. 1. Pattern for Distance Bounding Protocols

and a *validation phase*. Only the measurement phase is time-critical. The prover makes computationally inexpensive operations, such as *XOR*, during this phase and may use more sophisticated cryptographic algorithms, such as commitment schemes and message-authentication codes, in the other phases.

In [4], Meadows et al. present a suite of distance bounding protocols, following the pattern shown in Figure 1. Here, V denotes the verifier and P is the prover. Both parties initially create nonces NV and NP. V then sends a request to P, followed by a nonce NV. Upon receiving NV, P replies as quickly as possible with $F(NV, NP, P)$, where F is instantiated with an appropriate function. Finally P uses a key K_{VP} shared with V to create a message-authentication code (MAC). This proves that the nonce NP originated with P and binds the reply in the measurement phase to P's identity.

This protocol description is schematic in F. [4] provides four examples of instantiations of $F(NV, NP, P)$ built from different combinations of concatenation, exclusive-or, and hashing, e.g. $(NV \oplus P, NP)$ or, even simpler, (NV, NP, P). Each instantiation uses only simple cryptographic operations, which could even be implemented in hardware to further reduce their computation time.

The security property we want to prove is: "If V has successfully finished a protocol run with P, then V's conclusion about the distance to P is an upper bound on the physical distance between the two nodes." We will formalize this property, along with associated provisos, in subsequent sections.

3 Formal Model

In this section, we present our model of physical protocols. To support the verification of multiple protocols, we use locales to parameterize our model both with respect to the concrete protocol and message theory. Figure 2 depicts the theories we formalized in Isabelle and their dependencies. Some of these theories are concrete to begin with (e.g. *Geometric Properties of* $\mathbb{R}^3$) whereas other theories consist of locales or their interpretations. For example, the *Abstract Message Theory*

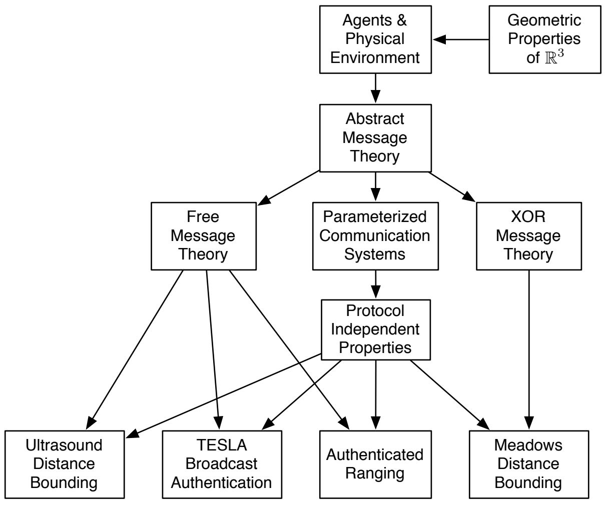
Agents & Physical Environment
Geometric Properties of $\mathbb{R}^3$
Abstract Message Theory
Free Message Theory
Parameterized Communication Systems
XOR Message Theory
Protocol Independent Properties
Ultrasound Distance Bounding
TESLA Broadcast Authentication
Authenticated Ranging
Meadows Distance Bounding

Fig. 2. Dependency Graph of our Isabelle Theory Files

contains a locale describing message theories, which is interpreted in our two concrete message theories (*Free* and *XOR*). In the theory *Parametrized Communication Systems*, we abstractly define the set of valid traces as a set of (parametric) inductive rules. In formalizations of concrete protocols using either of the two concrete message theories, we can therefore use both message-theory independent and message-theory specific facts by importing the required theories.

3.1 Agents and Environment

Agents are either honest agents or dishonest intruders. We model each kind using the natural numbers *nat*. Hence there are infinitely many agents of each kind.

$$\textbf{datatype}\ agent = \mathsf{Honest}\ nat \mid \mathsf{Intruder}\ nat$$

We refer to agents using capital letters like A and B. We also write H_A and H_B for honest agents and I_A and I_B for intruders, when we require this distinction. In contrast to the Dolev-Yao setting, agents' communication abilities are subject to the network topology and physical laws. Therefore, we cannot reduce a set of dishonest users at different locations to a single one.

Location and Physical Distance. To support reasoning about physical protocols, we associate every node A with a location loc_A. We define $loc : agent \rightarrow \mathbb{R}^3$ as an uninterpreted function constant. Protocol-specific assumptions about the position of nodes can be added as local assumptions to the corresponding theorems or using Isabelle/HOL's specification mechanism.[1] We use the standard

[1] Definition by specification allows us to assert properties of an uninterpreted function. It uses Hilbert's ϵ-operator and requires a proof that a function with the required properties exists.

Euclidean metric on $\mathbb{R}^3$ to define the physical distance between two agents A and B as $|\, loc_A - loc_B \,|$.

Taking the straight-line distance between the locations of the agents A and B in $\mathbb{R}^3$ as the shortest path (taken for example by electromagnetic waves when there are no obstacles), we define the line-of-sight communication distance $cdist_{LoS} : agent \times agent \to \mathbb{R}$ as

$$cdist_{LoS}(A, B) = \frac{|\, loc_A - loc_B \,|}{c} \,,$$

where c is the speed of light. Note that $cdist_{LoS}$ depends only on A and B's location and is independent of the network topology.

Transmitters, Receivers, and Communication Distance. To distinguish communication technologies with different characteristics, we equip each agent with an indexed set of transmitters.

$$\textbf{datatype}\ \mathit{transmitter} = \mathsf{Tx}\ \mathit{agent}\ \mathit{nat}$$

The constructor Tx returns a transmitter, given an agent A and an index i, denoted Tx^i_A. Receivers are formalized analogously.

$$\textbf{datatype}\ \mathit{receiver} = \mathsf{Rx}\ \mathit{agent}\ \mathit{nat}$$

We model the network topology using the uninterpreted function constant $cdist_{Net} : transmitter \times receiver \to \mathbb{R}_{\geq 0} \cup \{\perp\}$. We use $cdist_{Net}(Tx^i_A, Rx^j_B) = \perp$ to denote that Rx^j_B cannot receive transmissions from Tx^i_A. In contrast, $cdist_{Net}(Tx^i_A, Rx^j_B) = t$, where $t \neq \perp$, describes that Rx^j_B can receive signals (messages) emitted by Tx^i_A after a delay of at least t time units. This function models the minimal signal-transmission time for the given configuration. The time delay reflects environmental factors such as the communication medium used by the given transceivers and obstacles between transmitters and receivers. Since we assume that information cannot travel faster than the speed of light, we always require that $cdist_{LoS}(A, B) \leq cdist_{Net}(Tx^i_A, Rx^j_B)$ using the specification mechanism.

We use the formalization of real numbers and vectors provided in Isabelle's standard library for time and location. Additionally, we use the formalization of the Cauchy-Schwarz inequality [12] to establish that $cdist_{LoS}$ is a pseudometric.

3.2 Messages

Instead of restricting our model to a concrete message theory, we first define a locale that specifies a collection of message operators and their properties. In the context of this locale, we prove a number of properties independent of the protocol and message theory. For example, $cdist_{LoS}(A, B)$ is a lower bound on the time required for a nonce freshly created by A to become known by another agent B, since the nonce must be transmitted. For the results in [6], we have instantiated the locale with a message theory similar to Paulson's [7], modeling a free term algebra. In Section 3.6, we describe the instantiation with a message theory that includes the algebraic properties of the *XOR* operator, which we use in Section 4.

The theory of keys is shared by all concrete message theories and reuses Paulson's formalization. Keys are represented by natural numbers. The function $inv : key \to key$ partitions the set of keys into symmetric keys, where $inv\ k = k$, and asymmetric keys. We model key distributions as functions from agents to keys, e.g. the theory assumes that K_{AB} returns a shared symmetric key for a pair of agents A and B.

Abstract Message Theory Our MESSAGE_THEORY locale is parametric in the message type *'msg* and consists of the following function constants.

$\mathit{Nonce} : \mathit{agent} \to \mathit{nat} \to \mathit{'msg}$	$\mathit{Key} : \mathit{key} \to \mathit{'msg}$
$\mathit{Int} : \mathit{int} \to \mathit{'msg}$	$\mathit{Real} : \mathit{real} \to \mathit{'msg}$
$\mathit{Hash} : \mathit{'msg} \to \mathit{'msg}$	$\mathit{Crypt} : \mathit{key} \to \mathit{'msg} \to \mathit{'msg}$
$\mathit{MPair} : \mathit{'msg} \to \mathit{'msg} \to \mathit{'msg}$	$\mathit{parts} : \mathit{'msg\ set} \to \mathit{'msg\ set}$
$\mathit{subterms} : \mathit{'msg\ set} \to \mathit{'msg\ set}$	$\mathit{dm} : \mathit{agent} \to \mathit{'msg\ set} \to \mathit{'msg\ set}$

This formalizes that every interpretation of the MESSAGE_THEORY locale defines the seven given message construction functions and three functions on message sets. A *Nonce* is tagged with a unique identifier and the name of the agent who created it. This ensures that independently created nonces never collide. Indeed, even colluding intruders must communicate to share a nonce. The constructor *Crypt* denotes signing, asymmetric, or symmetric encryption, depending on the key used. We also require that functions for pairing (*MPair*), hashing (*Hash*), integers (*Int*), and reals (*Real*) are defined. We use the abbreviations $\langle A, B\rangle$ for $\mathit{MPair}\ A\ B$ and $\{m\}_k$ for $\mathit{Crypt}\, k\ m$. Moreover, we define $\mathit{MAC}_k(m) = \mathit{Hash}\langle \mathit{Key}\, k, m\rangle$ as the keyed MAC of the message m and $\mathit{MACM}_k(m) = \langle \mathit{MAC}_k(m), m\rangle$ as the pair consisting of m and its MAC. Additionally, every interpretation of MESSAGE_THEORY must define the functions *subterms*, *parts*, and *dm*. These respectively formalize the notions of subterms, extractable subterms, and the set of messages derivable from a set of known messages by a given agent. In the free message theory, *subterms* corresponds to syntactic subterms, for example $x \in \mathit{subterms}(\{\mathit{Hash}\, x\})$ while $x \notin \mathit{parts}(\{\mathit{Hash}\, x\})$.

We assume that the following properties hold for any interpretation of *parts*.

$$\frac{X \in H}{X \in \mathit{parts}(H)} \qquad \frac{X \in \mathit{parts}(H)}{\exists Y \in H. X \in \mathit{parts}(\{Y\})} \qquad \frac{G \subseteq H}{\mathit{parts}(G) \subseteq \mathit{parts}(H)}$$

$$\mathit{parts}(\mathit{parts}(H)) = \mathit{parts}(H) \qquad \mathit{parts}(H) \subseteq \mathit{subterms}(H)$$

These properties allow us to derive most of the lemmas about *parts* from Paulson's formalization [7] in our abstract setting. For example,

$$\mathit{parts}(G) \cup \mathit{parts}(H) = \mathit{parts}(G \cup H)\,.$$

Similar properties are assumed to hold for the *subterms* function.

We also assume properties of the message-derivation operator *dm* that state that no agent can guess another agent's nonces or keys, or forge encryptions or *MAC*s. These assumptions are reasonable for message theories formalizing idealized encryption.

$$\frac{\mathit{Nonce}\,B\ N_B \in \mathit{subterms}(\mathit{dm}\ A\ H) \quad A \neq B}{\mathit{Nonce}\ \ B\ N_B \in \mathit{subterms}(H)} \qquad \frac{\mathit{Key}\,k \in \mathit{parts}(\mathit{dm}\ A\ H)}{\mathit{Key}\,k \in \mathit{parts}(H)}$$

$$\frac{\{m\}_k \in \mathit{subterms}(\mathit{dm}\ A\ H)}{\{m\}_k \in \mathit{subterms}(H) \vee \mathit{Key}\,k \in \mathit{parts}(H)}$$

$$\frac{\mathit{MAC}_k(m) \in \mathit{subterms}(\mathit{dm}\ A\ H)}{\mathit{MAC}_k(m) \in \mathit{subterms}(H) \vee \mathit{Key}\,k \in \mathit{parts}(H)}$$

3.3 Events and Traces

We distinguish between three types of events: an agent sending a message, receiving a message, or making a claim. We use a polymorphic data type to model these different message types.

$$\textbf{datatype}\ \ '\mathit{msg}\ \mathit{event} = \mathsf{Send}\ \mathit{transmitter}\ \ '\mathit{msg}\ ('\mathit{msg}\ \mathit{list}) \\ \mid \mathsf{Recv}\ \mathit{receiver}\ \ '\mathit{msg}\ \mid \mathsf{Claim}\ \mathit{agent}\ \ '\mathit{msg}$$

A *trace* is a list of timed events, where a timed event $(t, e) \in \mathit{real} \times \mathit{event}$ pairs a time-stamp with an event.

A timed event $(t^S, \mathit{Send}\ \ \mathit{Tx}_A^i\ \ m\ L)$ denotes that the agent A has sent the message m using his transmitter Tx_A^i at time t^S and has associated the protocol data L with the event. The list of messages L models local state information and contains the messages used to construct m. The sender may require these messages in subsequent protocol steps. Storing L with the *Send* event is necessary since we support non-free message construction functions like *XOR* where a function's arguments cannot be recovered from the function's image alone.

A send event like the above may result in multiple timed *Recv*-events of the form $(t^R, \mathit{Recv}\ \mathit{Rx}_B^j\ m)$, where the time-stamps t^R and the receivers Rx_B^j must be consistent with the network topology. Note that the protocol data stored in L when sending the message does not affect the events on the receiver's side.

A *Claim*-event models a belief or conclusion made by a protocol participant, formalized as a message. For example, after successfully completing a run of a distance bounding protocol with a prover P, the verifier V concludes at time t that d is an upper bound on the distance to P. We model this by adding the timed event $(t, \mathit{Claim}\ V\ \langle P, \mathit{Real}\,d\rangle)$ to the trace. The protocol is secure if the conclusion holds for all traces containing this claim event.

Note that the time-stamps used in traces and the rules use the notion of absolute time. However, agents' clocks may deviate arbitrarily from absolute time. We must therefore translate the absolute time-stamps to model the local views of agents. We describe this translation in Section 3.4.

$$\frac{tr \in Tr \quad t^R \geq maxtime(tr) \quad (t^S, Send\ \ Tx_A^i\ \ m\ L) \in tr \quad cdist_{Net}(Tx_A^i, Rx_B^j) = t_{AB} \quad t_{AB} \neq \bot \quad t^R \geq t^S + t_{AB}}{tr.(t^R, Recv\ \ Rx_B^j\ \ m) \in Tr}\ \textsc{Net}$$

$$\frac{tr \in Tr \quad t \geq maxtime(tr) \quad m \in dm_{I_A}(knows_{I_A}(tr))}{tr.(t, Send\ \ Tx_{I_A}^k\ \ m\ []) \in Tr}\ \textsc{Fake}$$

$$\frac{}{[] \in Tr}\ \textsc{Nil}$$

$$\frac{tr \in Tr \quad t \geq maxtime(tr) \quad step \in proto \quad (act, m) \in step(view(H_A, tr), H_A, ctime(H_A, t)) \quad m \in dm_{H_A}(knows_{H_A}(tr))}{tr.(t, translateEv(H_A, act, m)) \in Tr}\ \textsc{Proto}$$

Fig. 3. Rules for Tr

Knowledge and Used Messages. Each agent A initially possesses some knowledge, denoted $initKnows_A$, which depends on the protocol executed. We use locales to underspecify the initial knowledge. We define a locale InitKnows that only includes the constant $initKnows : agent \to {}'msg\ \ set$. Different key distributions are specified by locales extending InitKnows with additional assumptions. For example, the locale InitKnows_Shared assumes that any two agents A and B share a secret key $Key\ \ K_{AB}$. In a system run with trace tr, A's knowledge consists of all messages he received together with his initial knowledge.

$$knows_A(tr) = \{m \mid \exists\ k\ t.(t,\ \ Recv\ \ Tx_A^k\ \ m) \in tr\} \cup\ initKnows_A$$

Each agent can derive all messages in the set $dm_A(knows_A(tr))$ by applying the derivation operator to the set of known messages. We use the *subterms* function to define the set of messages used in a trace tr.

$$used(tr) = \{n \mid \exists\ A\ k\ t\ m\ L.(t, Send\ \ Tx_A^k\ \ m\ L) \in tr\ \wedge\ n \in subterms(\{m\})\}$$

A nonce is *fresh* for a trace tr if it is not in $used(tr)$. Note that since a nonce is not fresh if its hash has been sent, we cannot use *parts* instead of *subterms* in the above definition.

3.4 Network, Intruder, and Protocols

We now describe the rules used to inductively define the set of traces Tr for a system parameterized by a protocol *proto*, an initial knowledge function *initKnows*, and the parameters from the abstract message theory. The base case, modeled by the Nil rule in Figure 3, states that the empty trace is a valid trace for all protocols. The other rules describe how valid traces can be extended. The rules model the network behavior, the possible actions of the intruders, and the actions taken by honest agents following the protocol steps.

Network Rule. The NET-rule models message transmission from transmitters to receivers, constrained by the network topology as given by $cdist_{Net}$. A *Send*-event from a transmitter may induce a *Recv*-event at a receiver only if the receiver can receive messages from the transmitter as specified by $cdist_{Net}$. The time delay between these events is bounded below by the communication distance between the transmitter and the receiver.

If there is a *Send*-event in the trace tr and the NET-rule's premises are fulfilled, a corresponding *Recv*-event is appended (denoted by $xs.x$) to the trace. The restriction on connectivity and transmission delay are ensured by $t_{AB} \neq \perp$ and $t^R \geq t^S + t_{AB}$. Here, t_{AB} is the communication distance between the receiver and transmitter, t^S is the sending time, and t^R is the receiving time.

Note that one *Send*-event can result in multiple *Recv*-events at the same receiver at different times. This is because $cdist_{Net}$ models the minimal communication distance and messages may also arrive later, for example due to the reflection of the signal carrying the message. Moreover, a *Send*-event can result in multiple *Recv*-events at different receivers, modeling for example broadcast communication. Finally, note that transmission failures and jamming by an intruder, resulting in message loss, are captured by not applying the NET-rule for a given *Send*-event and receiver, even if all premises are fulfilled.

The time-stamps associated with *Send*-events and *Recv*-events denote the starting times of message transmission and reception. Thus, our network rule captures the latency of links, but not the message-transmission time, which also depends on the message's size and the transmission speed of the transmitter and the receiver. Some implementation-specific attacks, such as those described in [5, 13], are therefore not captured in our model.

The premise $t \geq maxtime(tr)$, included in every rule (except NIL), ensures that time-stamps increase monotonically within each trace. Here t denotes the time-stamp associated with the new event and $maxtime(tr)$ denotes the latest time-stamp in the trace tr. This premise guarantees that the partial order on events induced by their time-stamps (note that events can happen simultaneously) is consistent with the order of events in the list representing the trace.

Intruder Rule. The FAKE-rule in Figure 3 describes the intruders' behavior. An intruder can always send any message m derivable from his knowledge. Intruders do not need any protocol state since they behave arbitrarily.

Since knowledge is distributed, we use explicit *Send*-events and *Recv*-events to model the exchange of information between colluding intruders. With an appropriate $cdist_{Net}$ function, it is possible to model an environment where the intruders are connected by high-speed links, allowing them to carry out wormhole attacks. Restrictions on the degree of cooperation between intruders can be modeled as predicates on traces. Internal and external attackers are both captured since they differ only in their initial knowledge (or associated transceivers).

Protocols. In contrast to intruders who can send arbitrary derivable messages, honest agents follow the protocol. A protocol is defined by a set of step functions.

Each step function takes the local view and time of an agent as input and returns all possible actions consistent with the protocol specification.

There are two types of possible actions, which model an agent either sending a message with a given transmitter id and storing the associated protocol data or making a claim.

$$\textbf{datatype}\ \textit{'msg action} = \ \mathsf{SendA}\ \textit{nat}\ (\textit{'msg list}\ )\ |\ \mathsf{ClaimA}$$

Note that message reception has already been modeled by the Net-rule.

An *action* associated with an agent and a message can be translated into the corresponding trace event using the *translateEv* function.

$$\begin{aligned} \mathit{translateEv}(A, \mathsf{SendA}\ \ k\ L, m) &= \mathit{Send}\ \ \mathit{Tx}_A^k\ \ m\ L \\ \mathit{translateEv}(A, \mathsf{ClaimA}\ , m) &= \mathit{Claim}\ A\ m \end{aligned}$$

A protocol *step* is therefore of type $\mathit{agent} \times \mathit{trace} \times \mathit{real} \rightarrow (\mathit{action} \times \mathit{msg})\ \mathit{set}$. Since our protocol rule Proto (described below) is parameterized by the protocol, we define a locale Protocol that defines a constant *proto* of type *step set* and inductively define *Tr* in the context of this locale.

Since the actions of an agent A only depend on his own previous actions and observations, we define A's view of a trace *tr* as the projection of *tr* on those events involving A. For this purpose, we introduce the function *occursAt*, which maps events to associated agents, e.g. $\mathit{occursAt}(\mathit{Send}\ \ \mathit{Tx}_A^i\ \ m\ L) = A$.

$$\mathit{view}(A, \mathit{tr}) = [(\mathit{ctime}(A, t), \mathit{ev})\ |(t, \mathit{ev}) \in \mathit{tr} \wedge \mathit{occursAt}(\mathit{ev}) = A]$$

Since the time-stamps of trace events refer to absolute time, the *view* function accounts for the offset of A's clock by translating times using the *ctime* function. Given an agent and an absolute time-stamp, the uninterpreted function $\mathit{ctime} : \mathit{agent} \times \mathit{real} \rightarrow \mathit{real}$ returns the corresponding time-stamp for the agent's clock.

Using the above definitions, we define the Proto-rule in Figure 3. For a given protocol, specified as a set of the step functions, the Proto rule describes all possible actions of honest agents, given their local views of a valid trace *tr* at a given time t. If all premises are met, the Proto-rule appends the translated event to the trace. Note that agents' behavior, modeled by the function *step*, is based only on the local clocks of the agents, i.e., agents cannot access the global time. Moreover, the restriction that all messages must be in $dm_{H_A}(knows_{H_A}(tr))$ ensures that agents only send messages derivable from their knowledge.

3.5 Protocol-independent Results

The set of all possible traces Tr is inductively defined by the rules Nil, Net, Fake, and Proto in the context of the Message_Theory, InitKnows, and Protocol locales. To verify a concrete protocol, we instantiate these locales thereby defining the concrete set of traces for the given protocol, initial knowledge, and message theory. Additional requirements are specified by defining new locales that extend Protocol and InitKnows.

Our first lemma specifies a lower bound on the time between when an agent first uses a nonce and another agent later uses the same nonce. The lemma holds whenever the initial knowledge of all agents does not contain any nonces.

Lemma 3.1 *Let A be an arbitrary (honest or dishonest) agent, N an arbitrary nonce, and $(t_A^S, Send \;\; Tx_A^i \;\; m_A \; L_A)$ the first event in a trace tr where $N \in subterms \; \{m_A\}$. If tr contains an event $(t, Send \;\; Tx_B^j \; m_B \; L_B)$ or $(t, Recv \;\; Rx_B^j \; m_B)$ where $A \neq B$ and $N \in subterms \; \{m_B\}$, then $t - t_A^S \geq cdist_{LoS}(A, B)$.*

Our next lemma holds whenever agents' keys are not *parts* of protocol messages and concerns when MACs can be created. Note that we need the notion of extractable subterms here since protocols use keys in MACs, but never send them in extractable positions.

Lemma 3.2 *Let A and B be honest agents and C a different possibly dishonest agent. Furthermore let $(t_C^S, Send \;\; Tx_C^i \;\; m_C \;\; L_C)$ be an event in the trace tr where $MAC_{K_{AB}}(m) \in subterms \; \{m_C\}$ for some message m and a shared secret key K_{AB}. Then, for E either equal to A or B, there is a send event $(t_E^S, Send \;\; Tx_E^j \;\; m_E \;\; L_E) \in tr$ where $MAC_{K_{AB}}(m) \in subterms \; \{m_E\}$ and $t_C^S - t_E^S \geq cdist_{LoS}(E, C)$.*

Note that the lemmas are similar to the axioms presented in [4]. The proofs of these lemmas can be found in our Isabelle/HOL formalization [14].

3.6 XOR Message Theory

In this section, we present a message theory including *XOR*, which instantiates the message-theory locale introduced in Section 3.2.

The Free Message Type. We first define the free term algebra of messages. Messages are built from agent names, integers, reals, nonces, keys, hashes, pairs, encryption, exclusive-or, and zero.

$$
\begin{aligned}
\textbf{datatype}\; \mathit{fmsg} = {} & \mathsf{AGENT}\; \mathit{agent} \mid \mathsf{INT}\; \mathit{int} \mid \mathsf{REAL}\; \mathit{real} \\
& \mid \mathsf{NONCE}\; \mathit{agent\; nat} \mid \mathsf{KEY}\; \mathit{key} \mid \mathsf{HASH}\; \mathit{fmsg} \\
& \mid \mathsf{MPAIR}\; \mathit{fmsg\; fmsg} \mid \mathsf{CRYPT}\; \mathit{key\; fmsg} \\
& \mid \mathit{fmsg} \;\bar{\oplus}\; \mathit{fmsg} \mid \mathsf{ZERO}
\end{aligned}
$$

To faithfully model $\bar{\oplus}$, we require the following set of equations E:

$$
\begin{array}{rlrl}
(x \bar{\oplus} y) \bar{\oplus} z \approx x \bar{\oplus} (y \bar{\oplus} z) & (\mathsf{A}) & x \bar{\oplus} y \approx y \bar{\oplus} x & (\mathsf{C}) \\
x \bar{\oplus} \mathsf{ZERO} \approx x & (\mathsf{U}) & x \bar{\oplus} x \approx \mathsf{ZERO} & (\mathsf{N})
\end{array}
$$

We define the corresponding equivalence relation $=_E$ as the reflexive, symmetric, transitive, and congruent closure of E. We also define the reduction relation $\rightarrow_E$ as the reflexive, transitive, and congruent closure of E, where the cancellation rules (U) and (N) are directed from left to right and (A) and (C) can be used in both directions. Note that $x \rightarrow_E y$ implies $x =_E y$, for all x and y.

$$\frac{}{\mathit{reduced}\ (\mathsf{AGENT}\ \ a)}\ \textsc{Agent} \qquad \frac{}{\mathit{reduced}\ (\mathsf{INT}\ \ i)}\ \textsc{Int} \qquad \frac{}{\mathit{reduced}\ (\mathsf{REAL}\ \ i)}\ \textsc{Real}$$

$$\frac{}{\mathit{reduced}\ (\mathsf{NONCE}\ \ a\ na)}\ \textsc{Nonce} \qquad \frac{\mathit{reduced}\ h}{\mathit{reduced}\ (\mathsf{HASH}\ \ h)}\ \textsc{Hash}$$

$$\frac{\mathit{reduced}\ a \quad \mathit{reduced}\ b}{\mathit{reduced}\ (\mathsf{MPAIR}\ \ a\ b)}\ \textsc{MPair} \qquad \frac{\mathit{reduced}\ m}{\mathit{reduced}\ (\mathsf{CRYPT}\ \ k\ m)}\ \textsc{Crypt}$$

$$\frac{\mathit{reduced}\ a \quad \mathit{reduced}\ b \quad \mathit{standard}\ a \quad a < \mathit{first}\ b \quad b \neq \mathsf{ZERO}}{\mathit{reduced}\ (a \mathbin{\bar{\oplus}} b)}\ \textsc{Xor}$$

Fig. 4. Rules for *reduced*

Reduced Messages and the Reduction Function. We define the predicate *standard* on *fmsg* that returns true for all messages where the outermost constructor is neither equal to $\bar{\oplus}$ nor *ZERO*. We also define the projection function *first*, where *first* x equals a when $x = a \mathbin{\bar{\oplus}} b$ for some a and b and equals x otherwise. We use both functions to define *reduced* messages. We show below that every equivalence class with respect to $=_E$ contains exactly one *reduced* message, used as the classes canonical representative. To handle commutativity, we define a linear order on *fmsg* using the underlying orders on *nat*, *int*, and *agent*. A message is *reduced* if $\bar{\oplus}$ messages are right-associated, ordered, and all cancellation rules have been applied. This is captured by the inductive definition in Figure 4.

To obtain a decision procedure for $x =_E y$, we define a reduction function $\downarrow$ on *fmsg* that reduces a message, that is $x{\downarrow}$ is reduced and $(x{\downarrow}) =_E x$. We begin with the definition of an auxiliary function $\oplus^? : \mathit{fmsg} \to \mathit{fmsg}$: $a \oplus^? b = \mathsf{if}\ b = \mathsf{ZERO}\ \mathsf{then}\ a\ \mathsf{else}\ a \mathbin{\bar{\oplus}} b$.

We now define the main part of the reduction: the function $\oplus^{\downarrow} : \mathit{fmsg} \to \mathit{fmsg} \to \mathit{fmsg}$ presented in Figure 5 combines two reduced messages a and b, yielding $(a \mathbin{\bar{\oplus}} b){\downarrow}$. Note that the order of the equations is relevant: given overlapping patterns, the first applicable equation is used. The algorithm is similar to a merge-sort on lists. The first two cases are straightforward and correspond to the application of the (U) rule. The other cases are justified by combinations of all four rules.

The definition of $(\cdot){\downarrow} : \mathit{fmsg} \to \mathit{fmsg}$ is straightforward:

$$\begin{aligned}
(\mathsf{HASH}\ \ m){\downarrow} &= \mathsf{HASH}\ \ m{\downarrow}\\
(\mathsf{MPAIR}\ \ a\ b){\downarrow} &= \mathsf{MPAIR}\ \ (a{\downarrow})\ (b{\downarrow})\\
(\mathsf{CRYPT}\ \ k\ m){\downarrow} &= \mathsf{CRYPT}\ \ k\ (m{\downarrow})\\
(a \mathbin{\bar{\oplus}} b){\downarrow} &= (a{\downarrow}) \oplus^{\downarrow} (b{\downarrow})\\
x{\downarrow} &= x
\end{aligned}$$

$$
\begin{array}{lll}
x \oplus^{\downarrow} \mathsf{ZERO} & = x & (1)\\
\mathsf{ZERO} \oplus^{\downarrow} x & = x & (2)\\
(a_1 \bar{\oplus} a_2) \oplus^{\downarrow} (b_1 \bar{\oplus} b_2) & = \mathsf{if}\ a_1 = b_1\ \mathsf{then}\ a_2 \oplus^{\downarrow} b_2 & (3)\\
 & \quad \mathsf{else\ if}\ a_1 < b_1\ \mathsf{then}\ a_1 \oplus^{?} (a_2 \oplus^{\downarrow} (b_1 \bar{\oplus} b_2)) & (4)\\
 & \quad \mathsf{else}\ b_1 \oplus^{?} ((a_1 \bar{\oplus} a_2) \oplus^{\downarrow} b_2) & (5)\\
(a_1 \bar{\oplus} a_2) \oplus^{\downarrow} b & = \mathsf{if}\ a_1 = b\ \mathsf{then}\ a_2 & (6)\\
 & \quad \mathsf{else\ if}\ a_1 < b\ \mathsf{then}\ a_1 \oplus^{?} (a_2 \oplus^{\downarrow} b) & (7)\\
 & \quad \mathsf{else}\ b \bar{\oplus} (a_1 \bar{\oplus} a_2) & (8)\\
a \oplus^{\downarrow} (b_1 \bar{\oplus} b_2) & = (b_1 \bar{\oplus} b_2) \oplus^{\downarrow} a & (9)\\
a \oplus^{\downarrow} b & = \mathsf{if}\ a = b\ \mathsf{then}\ \mathsf{ZERO} & (10)\\
 & \quad \mathsf{else\ if}\ a < b\ \mathsf{then}\ a \bar{\oplus} b\ \mathsf{else}\ b \bar{\oplus} a & (11)
\end{array}
$$

Fig. 5. Definition of $\oplus^{\downarrow}$

We have proved the following facts about reduction: (1) if *reduced* x then $(x{\downarrow}) = x$, (2) *reduced* $(x{\downarrow})$, and (3) $x \rightarrow_E (x{\downarrow})$. Using these facts we establish:

Lemma 3.3 *For all messages x and y, $x =_E y$ iff (if and only if) $(x{\downarrow}) = (y{\downarrow})$. Furthermore, if reduced x and reduced y, then $x =_E y$ iff $x = y$.*

The Message Type, parts, and dm. Given the above lemma, we use the function $\downarrow$ and the predicate *reduced* to characterize $=_E$. Isabelle's *typedef* mechanism allows us to define the quotient type *msg* with $\{m \mid reduced\ m\}$ as the representing set. This defines a new type *msg* with a bijection between the representing set in *fmsg* and *msg* given by the function $Abs\_msg : fmsg \rightarrow msg$ and its inverse $Rep\_msg : msg \rightarrow fmsg$. Note that $=_E$ on *fmsg* corresponds to object-logic equality on *msg*. This is reflected in the following lemma.

Lemma 3.4 *For all messages x and y, $x = y$ iff $Rep\_msg(x) =_E Rep\_msg(y)$.*

We define functions on *msg* by using the corresponding definitions on *fmsg* and the embedding and projection functions. That is, we lift the message constructors to *msg* using the $\downarrow$ function. For example:

$$
\begin{array}{ll}
Nonce\ a\ n = & Abs\_msg(\mathsf{NONCE}\ a\ n)\\
MPair\ a\ b = & Abs\_msg(\mathsf{MPAIR}\ (Rep\_msg(a))\ (Rep\_msg(b)))\\
Hash\ m \quad = & Abs\_msg(\mathsf{HASH}\ (Rep\_msg(m)))\\
Xor\ a\ b \quad = & Abs\_msg((Rep\_msg(a) \bar{\oplus} Rep\_msg(b)){\downarrow})\\
Zero \qquad = & Abs\_msg(\mathsf{ZERO}\ )
\end{array}
$$

In the following, we write 0 for *Zero* and $x \oplus y$ for *Xor* $x\ y$. We define a function *fparts* on *fmsg* that returns all extractable subterms of a given message, e.g.

$$\frac{m \in M}{m \in dm_A(M)}\ \text{INJ} \qquad \frac{}{0 \in dm_A(M)}\ \text{ZERO} \qquad \frac{m \in dm_A(M)}{\mathit{Hash}\ m \in dm_A(M)}\ \text{HASH}$$

$$\frac{\langle m, n\rangle \in dm_A(M)}{m \in dm_A(M)}\ \text{FST} \qquad \frac{\langle m, n\rangle \in dm_A(M)}{n \in dm_A(M)}\ \text{SND}$$

$$\frac{m \in dm_A(M) \qquad n \in dm_A(M)}{\langle m, n\rangle \in dm_A(M)}\ \text{PAIR} \qquad \frac{m \in dm_A(M) \qquad n \in dm_A(M)}{m \oplus n \in dm_A(M)}\ \text{XOR}$$

$$\frac{m \in dm_A(M) \qquad \mathit{Key}\ k \in DM_A(M)}{\{m\}_k \in dm_A(M)}\ \text{ENC} \qquad \frac{}{\mathit{Nonce}\ A\, n \in dm_A(M)}\ \text{NONCE}$$

$$\frac{\{m\}_k \in dm_A(M) \qquad (\mathit{Key}\ k)^{-1} \in dm_A(M)}{m \in dm_A(M)}\ \text{DEC} \qquad \frac{}{\mathit{Agent}\ a \in dm_A(M)}\ \text{AGENT}$$

$$\frac{}{\mathit{Int}\ n \in dm_A(M)}\ \text{INT} \qquad \frac{}{\mathit{Real}\ n \in dm_A(M)}\ \text{REAL}$$

Fig. 6. Rules for $dm_A(M)$

$m \in \mathit{fparts}(\{\mathit{CRYPT}\ k\ m\})$, but $m \notin \mathit{fparts}(\{\mathit{HASH}\ m\})$. The function *parts* on *msg* that is used to instantiate the function of the same name in the message-theory locale is then defined as

$$\mathit{parts}(H) = \{\mathit{Abs\_msg}(m) \mid m \in \mathit{fparts}\ \{\mathit{Rep\_msg}(x) \mid x \in H\}\}\,.$$

This defines the *parts* of a message m in the equivalence class represented by $m{\downarrow}$ as the *fparts* of $m{\downarrow}$. For example $\mathit{parts}(\{X \oplus X\}) = \{0\}$. The function *subterms* is defined similarly, but returns all subterms and not just the extractable ones. We give the rules for the inductively-defined message-derivation operator $dm : \mathit{agent} \rightarrow \mathit{msg\ set} \rightarrow \mathit{msg\ set}$ in Figure 6. The rules specify message decryption, projection on pairs, pairing, encryption, signing, hashing, *XOR*ing, and the generation of integers, reals, agent names, and nonces. For example, the *Dec*-rule states that if an agent A can derive the ciphertext $\{m\}_k$ and the decryption key $(\mathit{Key}\ k)^{-1}$, then he can also derive the plaintext m. When $\mathit{Key}\ k$ is used as a signing key, A uses the verification key $(\mathit{Key}\ k)^{-1}$ to verify the signature. The Xor rule uses the constructor *Xor*, which ensures that the resulting message is reduced. We can now interpret the locale Message_Theory by proving that the defined operators have the required properties.

4 Protocol Correctness

In this section, we present highlights from our verification of instances of the protocol pattern in Figure 1. Complete details are provided in [14]. We first formalize the protocol pattern and its desired security properties. Afterwards, we reduce the security of pattern instances to properties of the rapid-response function used. Finally, we consider several concrete rapid-response functions proposed in [4] and analyze the security of the corresponding instantiations.

4.1 Protocol Rules

We conduct our security analysis using our *XOR* message theory. We first define the concrete initial knowledge as $\mathit{initKnows}_A = \bigcup_B \{\mathit{Key}\, K_{AB}\}$, which we interpret as an instance of the InitKnows_Shared locale. Next, we instantiate the Protocol locale by defining the protocol pattern in Figure 1 as $\mathit{proto} = \{\mathit{mdb}_1, \mathit{mdb}_2, \mathit{mdb}_3, \mathit{mdb}_4\}$. Each step function $\mathit{mdb}_i(A, tr, t)$ yields the possible actions of the agent A executing the protocol step i with his view of the trace tr at the local time t.

Our distance bounding protocol definition uses the uninterpreted rapid-response function $F : \mathit{msg} \times \mathit{msg} \times \mathit{msg} \rightarrow \mathit{msg}$ and consists of the following steps.[2]

Start: An honest verifier V can start a protocol run by sending a fresh nonce using his radio transmitter r at any local time t.

$$\frac{\mathit{Nonce}\, V\, \mathit{NV} \notin \mathit{used}(tr)}{(\mathit{SendA}\;\; r\; [\,], \mathit{Nonce}\, V\, \mathit{NV}) \in \mathit{mdb}_1(V, tr, t)}$$

Rapid-Response: If a prover P receives a nonce NV, he may continue the protocol by replying with the message $F(NV, NP, P)$ and storing the protocol data $[NV, \mathit{Nonce}\, P\, NP]$. This information must be stored since it is needed in the authentication step and cannot, in general, be reconstructed from $F(NV, NP, P)$.

$$\frac{(t_P^R, \mathit{Recv}\;\; \mathit{Rx}_P^r\;\; \mathit{NV}) \in tr \qquad \mathit{Nonce}\, P\, \mathit{NP} \notin \mathit{used}(tr)}{(\mathit{SendA}\;\; r\;\; [\mathit{NV}, \mathit{Nonce}\, P\, \mathit{NP}], F(\mathit{NV}, \mathit{Nonce}\, P\, \mathit{NP}, \mathit{Agent}\, P)) \in \mathit{mdb}_2(P, tr, t)}$$

Authentication: After a prover P has answered a verifier's challenge with a rapid-response, he authenticates the response with the corresponding MAC.

$$\frac{\begin{array}{c}(t_P^R, \mathit{Recv}\;\; \mathit{Rx}_P^r\;\; \mathit{NV}) \in tr \\ (t_P^S, \mathit{Send}\;\; \mathit{Tx}_P^r\;\; F(\mathit{NV}, \mathit{Nonce}\, P\, \mathit{NP}, \mathit{Agent}\, P)\; [\mathit{NV}, \mathit{Nonce}\, P\, \mathit{NP}]) \in tr\end{array}}{(\mathit{SendA}\;\; r\; [\,], \mathit{MACM}_{K_{VP}}(\mathit{NV}, \mathit{Nonce}\, P\, \mathit{NP}, \mathit{Agent}\;\; P)) \in \mathit{mdb}_3(P, tr, t)}$$

Claim: Suppose the verifier receives a rapid-response in the measurement phase at time t_1^R and the corresponding MAC in the validation phase, both involving the nonce that he initially sent at time t_1^S. The verifier therefore concludes that $(t_1^R - t_1^S) * c/2$ is an upper bound on the distance to the prover P, where c denotes the speed of light.

$$\frac{\begin{array}{c}(t_1^S, \mathit{Send}\;\; \mathit{Tx}_V^r\;\; (\mathit{Nonce}\, V\, \mathit{NV})\; [\,]) \in tr \\ (t_1^R, \mathit{Recv}\; (\mathit{Rx}_V^r)\; F(\mathit{Nonce}\, V\, \mathit{NV}, \mathit{NP}, \mathit{Agent}\;\; P)) \in tr \\ (t_2^R, \mathit{Recv}\;\; \mathit{Rx}_V^r\;\; \mathit{MACM}_{K_{VP}}(\mathit{Nonce}\, V\, \mathit{NV}, \mathit{NP}, \mathit{Agent}\, P)) \in tr\end{array}}{(\mathit{ClaimA}, (\mathit{Agent}\;\; P, \mathit{Real}\; (t_1^R - t_1^S) * c/2)) \in \mathit{mdb}_4(V, tr, t)}$$

[2] We have formalized each step in Isabelle/HOL using set comprehension, but present the steps here as rules for readability. For each rule r, the set we define by comprehension is equivalent to the set defined inductively by the rule r.

The set of traces *Tr* is inductively defined by the rules NIL, FAKE, NET, and PROTO. Note that the same set of traces can be inductively defined by the NIL, FAKE, and NET rules along with rules describing the individual protocol steps. See [6] for more details on these different representations.

4.2 Security Properties

In this section, we present properties of distance bounding protocols instantiating the protocol pattern from Figure 1. First, we note that we only consider honest verifiers. Since successful protocol execution leads to claims on the verifier's side, it makes no sense to consider dishonest verifiers. However, we distinguish between honest and dishonest provers. For honest provers, we require that the claimed distance after a successful protocol execution denoted by a *Claim* event is an upper bound on the physical distance between the prover and the verifier.

Definition 4.1 *A distance bounding protocol is secure for honest provers (hp-secure) iff whenever* $\mathit{Claim}\ V\ \langle P, \mathit{Real}\, d\rangle \in tr$*, then* $d \geq |\, loc_V - loc_P\,|$.

In the case of a dishonest prover, it is impossible to distinguish between different intruders who exchange their keys. Hence, our weaker property must accommodate for attacks where one intruder pretends to be another intruder. We therefore require that the claimed distance is an upper bound on the distance between the verifier and some intruder.

Definition 4.2 *A distance bounding protocol is secure for dishonest provers (dp-secure) iff whenever* $\mathit{Claim}\ V\ \langle P, \mathit{Real}\, d\rangle \in tr$ *for an intruder* P*, then there is an intruder* P' *such that* $d \geq |\, loc_V - loc_{P'}\,|$.

4.3 Security Proofs based on Properties of F

In order to prove security properties of an instance of the protocol pattern, we show how security properties of the protocol can be reduced to properties of F. To prove the security of a concrete instantiation, we must then only prove that it has the required properties.

Definition 4.3 *In the following, let* X*,* Y*, and* Z*, be messages,* m *an atomic message or a MAC,* A*,* B*, and* C*, agents, and NA, NB, and NC nonce tags. We define the following properties for a function* $F : msg \times msg \times msg \rightarrow msg$*:*

(P0)

(a) $\mathit{If}\ X \in H$*, then* $F(X, \mathit{Nonce}\ A\ NA, \mathit{Agent}\ A) \in dm_A\, H$.

(b) $\mathit{If}\ m \in parts(\{F(X,Y,Z)\})$*, then* $m \in parts(\{X,Y,Z\})$.

(c) $F(X, \mathit{Nonce}\ A\ NA, \mathit{Agent}\ A) \neq \mathit{Nonce}\ C\ NC$.

(d) $F(X, \mathit{Nonce}\ A\ NA, \mathit{Agent}\ A) \neq MACM_{K_{BC}}(Y, \mathit{Nonce}\ C\ NC, \mathit{Agent}\ C)$.

(P1) $\mathit{Nonce}\ A\ NA \in subterms(F(\mathit{Nonce}\ A\ NA, X, \mathit{Agent}\ B))$

(P2) $\mathit{Nonce}\ B\ NB \in subterms(F(X, \mathit{Nonce}\ B\ NB, \mathit{Agent}\ B))$

(P3) $\mathit{Agent}\ B \in subterms(F(\mathit{Nonce}\ A\ NA, X, \mathit{Agent}\ B))$

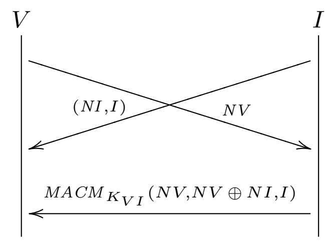

Fig. 7. Jumping the Gun Attack

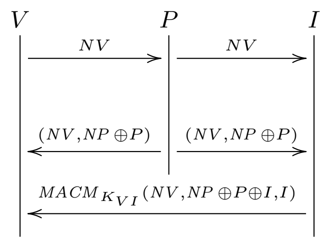

Fig. 8. Impersonation Attack

The property (P0) specifies well-formedness conditions on F: (a) F can be computed from the challenge, (b) F neither introduces new atomic messages nor MACs, which are not among its arguments, and (c)–(d) F cannot be confused with the other protocol messages. Properties (P1)–(P3) state that arguments in certain positions are always subterms of F's result as long as the remaining arguments have the required types. Using these properties, we prove the following lemmas that give sufficient conditions on F for *hp-security* and *dp-security*.

Theorem 4.4 *For every function F with properties (P0)–(P2), the resulting instance of the protocol pattern is an hp-secure distance bounding protocol.*

In the proof of Theorem 4.4, (P1) is used to ensure that the nonce sent as a challenge by the verifier must be involved in the computation of the prover's response. Analogously, (P2) ensures that the response can only be computed if the fresh nonce contributed by the prover is known.

For the case of a dishonest prover, we additionally require (P3) to prevent a dishonest prover from taking credit for a response sent by an honest one.

Theorem 4.5 *For every function F with properties (P0)–(P3), the resulting instance of the protocol pattern is a dp-secure distance bounding protocol.*

We have proved that $\langle NV \oplus P, NP\rangle$ and $\langle NV, NP, P\rangle$ have properties (P0)–(P3) and therefore the corresponding protocol instances are both *hp-secure* and *dp-secure.*

The function $F_1(NV, NP, P) = \langle NV \oplus NP, P\rangle$ lacks (P1) and is not *dp-secure.* To see that (P1) fails, consider $F_1(NV, NV, P) = \langle 0, P\rangle$, which does not contain NV as a subterm. Remember that we have defined the subterms of a message t as the subterms of $t{\downarrow}$. A dishonest prover I can use this to execute the "jumping the gun" attack given in Figure 7. The attack uses the equivalence $F_1(NV, NV \oplus NI, I) = \langle NV \oplus(NV \oplus NI), I\rangle = \langle NI, I\rangle$.

In contrast, the function $F_2(NV, NP, P) = \langle NV, NP \oplus P\rangle$ lacks property (P3) and is therefore *hp-secure* but not *dp-secure.* To see that (P3) fails, consider $F_2(NV, NI \oplus \mathit{Agent}\, P, \mathit{Agent}\, P) = \langle NV, NI\rangle$, which does not contain $\mathit{Agent}\, P$ as a subterm. This leads to the impersonation attack depicted in Figure 8 violating the *dp-security* property. This attack uses the equivalence $F_2(NV, NP, P) = \langle NV, NP \oplus P\rangle = F_2(NV, NP \oplus I \oplus P, I)$.

Overall, proving the properties (P0)–(P4) for a given function and applying Theorems 4.4 and 4.5 is much simpler than the corresponding direct proofs. However, finding the correct properties and proving these theorems for the *XOR* message theory turned out to be considerably harder than proofs for comparable theorems about a fixed protocol in the free message theory. This additional complexity mainly stems from the untyped protocol formalization necessary to realistically model the *XOR* operator.

5 Related Work and Conclusions

Our model of physical security protocols extends the Dolev-Yao model with dense time, network topology, and node location. Each of these properties has been handled individually in other approaches. For example, discrete and dense time have been used to reason about security protocols involving timestamps or timing issues like timeouts and retransmissions [15, 16]. Models encompassing network topology have been studied in the context of secure routing protocols for ad hoc networks [17, 18]. Node location and relative distance have been considered in [4, 5]. In [6], we compare our work with these approaches in more detail. While these approaches address individual physical properties, to the best of our knowledge our model is the first that provides a general foundation for reasoning about all three of these properties and their interplay.

The protocol pattern we study comes from Meadows et al. [4]. The authors give a condition on instances of F (called "simply stable") under which the resulting protocol is correct in the case of honest provers. They also investigate the two function instances F_1 and F_2 that we presented above. They give the attack on F_1. However, as they do not consider the possibility of dishonest provers in their proof, they classify F_2 as secure. Their correctness proofs are based on a specialized authentication logic, tailored for distance-bounding protocols, that is presented axiomatically. While they do not provide a semantics, we note that their axioms can be suitably interpreted and derived within our setting.

From the specification side, our work builds on several strands of research. The first is the modeling of security protocols as inductively-defined sets of traces. Our work is not only inspired by Paulson's inductive method [7], we were able to reuse some of his theories, in particular his key theory and much of his free message algebra.

The second strand is research on formalizing equational theories in theorem provers. Courant and Monin [19] formalize a message algebra including *XOR* in Coq, which they use to verify security APIs. They introduce an uninterpreted normalization function with axiomatically-defined properties, which in turn is used to define their equivalence relation. Since they do not use a quotient type to represent equivalence classes, they must account for different representations of equivalent messages. Paulson's work on defining functions on equivalence classes [20] uses a quotient type construction in Isabelle/HOL that is similar to ours, but represents equivalence classes as sets instead of canonical elements.

The final strand concerns developing reusable results by proving theorems about *families* of inductively-defined sets. In our work, we give generic formalizations of sets of messages and traces, including lemmas, which hold for different instances. The key idea is to use one parameterized protocol rule, instead of a collection of individual rules, for the protocol steps. The inductive definition is then packaged in a locale, where the locale parameter is the rule parameter, and locale assumptions formalize constraints on the protocol steps (such as well-formedness). Note that this idea is related to earlier work on structuring (meta)theory [21–23] using parameterized inductively-defined sets, where the theorems themselves directly formalize the families of sets. Overall, our work constitutes a substantial case study in using locales to structure reusable theories about protocols. Another case study, in the domain of linear arithmetic, is that of [24].

In conclusion, our model has enabled us to formalize protocols, security properties, and environmental assumptions that are not amenable to formal analysis using other existing approaches. As future work, we plan to extend our model to capture additional properties of wireless security protocols. We also intend to refine our model to capture message sizes and transmission rate, rapid bit exchange, and online guessing attacks, which would allow us to analyze protocols such as those presented in [1].

References

1. Brands, S., Chaum, D.: Distance-bounding protocols. Lecture Notes in Computer Science **765** (1994) 344–359
2. Capkun, S., Buttyan, L., Hubaux, J.P.: SECTOR: secure tracking of node encounters in multi-hop wireless networks. In: SASN '03: Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, New York, NY, USA, ACM Press (2003) 21–32
3. Hancke, G.P., Kuhn, M.G.: An RFID distance bounding protocol. In: SECURECOMM '05: Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks, Washington, DC, USA, IEEE Computer Society (2005) 67–73
4. Meadows, C., Poovendran, R., Pavlovic, D., Chang, L., Syverson, P.: Distance bounding protocols: Authentication logic analysis and collusion attacks. In: Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks. Springer-Verlag (2006) 279–298
5. Sastry, N., Shankar, U., Wagner, D.: Secure verification of location claims. In: WiSe '03: Proceedings of the 2003 ACM workshop on Wireless security, New York, NY, USA, ACM Press (2003) 1–10
6. Schaller, P., Schmidt, B., Basin, D., Capkun, S.: Modeling and verifying physical properties of security protocols for wireless networks. CSF-22: 22nd IEEE Computer Security Foundations Symposium (2009) To appear.
7. Paulson, L.C.: The inductive approach to verifying cryptographic protocols. Journal of Computer Security **6** (1998) 85–128
8. Nipkow, T., Paulson, L., Wenzel, M.: Isabelle/HOL: A Proof Assistant for Higher-Order Logic. Springer (2002)

9. Capkun, S., Hubaux, J.P.: Secure positioning of wireless devices with application to sensor networks. In: INFOCOM, IEEE (2005) 1917–1928
10. Perrig, A., Tygar, J.D.: Secure Broadcast Communication in Wired and Wireless Networks. Kluwer Academic Publishers, Norwell, MA, USA (2002)
11. Ballarin, C.: Interpretation of locales in Isabelle: Theories and proof contexts. Mathematical Knowledge Management (MKM 2006), LNAI **4108** (2006)
12. Porter, B.: Cauchy's mean theorem and the cauchy-schwarz inequality. The Archive of Formal Proofs (March 2006) Formal proof development.
13. Clulow, J., Hancke, G.P., Kuhn, M.G., Moore, T.: So near and yet so far: Distance-bounding attacks in wireless networks. In: Security and Privacy in Ad-hoc and Sensor Networks, Springer (2006) 83–97
14. Schmidt, B. and Schaller, P.: Isabelle Theory Files: Modeling and Verifying Physical Properties of Security Protocols for Wireless Networks. http://people.inf.ethz.ch/benschmi/ProtoVeriPhy/
15. Delzanno, G., Ganty, P.: Automatic Verification of Time Sensitive Cryptographic Protocols. TACAS '04: Proceedings of the 10th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (2004)
16. Evans, N., Schneider, S.: Analysing Time Dependent Security Properties in CSP Using PVS. In: ESORICS '00: Proceedings of the 6th European Symposium on Research in Computer Security, London, UK, Springer-Verlag (2000) 222–237
17. Acs, G., Buttyan, L., Vajda, I.: Provably Secure On-Demand Source Routing in Mobile Ad Hoc Networks. IEEE Transactions on Mobile Computing **5**(11) (2006) 1533–1546
18. Yang, S., Baras, J.S.: Modeling vulnerabilities of ad hoc routing protocols. In: SASN '03: Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks, New York, NY, USA, ACM (2003) 12–20
19. Courant, J., Monin, J.: Defending the bank with a proof assistant. In: Proceedings of the 6th International Workshop on Issues in the Theory of Security (WITS'06). (2006) 87–98
20. Paulson, L.: Defining functions on equivalence classes. ACM Transactions on Computational Logic **7**(4) (2006) 658–675
21. Basin, D., Constable, R.: Metalogical frameworks. In Huet, G., Plotkin, G., eds.: Logical Environments. Cambridge University Press (1993) 1–29 Also available as Technical Report MPI-I-92-205.
22. Basin, D., Matthews, S.: Logical frameworks. In Gabbay, D., Guenthner, F., eds.: Handbook of Philosophical Logic, second edition. Volume 9. Kluwer Academic Publishers, Dordrecht (2002) 89–164
23. Basin, D., Matthews, S.: Structuring metatheory on inductive definitions. Information and Computation **162**(1–2) (October/November 2000)
24. Nipkow, T.: Reflecting quantifier elimination for linear arithmetic. Formal Logical Methods for System Security and Correctness (2008) 245