A Multiset Rewriting Model for Specifying and Verifying Timing Aspects of Security Protocols

Musab A. Alturki[1,2] Tajana Ban Kirigin[3] Max Kanovich[4,8] Vivek Nigam[5,6] Andre Scedrov[7,8] and Carolyn Talcott[9]

[1] KFUPM, Dhahran, Saudi Arabia, musab@kfupm.edu.sa
[2] Runtime Verification Inc., USA
[3] University of Rijeka, Department of Mathematics, HR, bank@math.uniri.hr
[4] University College, London, UK, m.kanovich@ucl.ac.uk
[5] Federal University of Paraíba, João Pessoa, Brazil, vivek@ci.ufpb.br
[6] fortiss, Germany, nigam@fortiss.org
[7] University of Pennsylvania, Philadelphia, USA, scedrov@math.upenn.edu
[8] National Research University Higher School of Economics, Moscow, Russia
[9] SRI International, USA, clt@csl.sri.com

Abstract. Catherine Meadows has played an important role in the advancement of formal methods for protocol security verification. Her insights on the use of, for example, narrowing and rewriting logic has made possible the automated discovery of new attacks and the shaping of new protocols. Meadows has also investigated other security aspects, such as, distance bounding protocols and denial of service attacks. We have been greatly inspired by her work. This paper describes the use of Multiset Rewriting for the specification and verification of timing aspects of protocols, such as network delays, timeouts, timed intruder models and distance bounding properties. We detail these timed features with a number of examples and describe decidable fragments of related verification problems.

1 Introduction

Protocol security verification is one of the best success stories of formal methods. Indeed a number of attacks and corrections have been discovered since Lowe found an attack on the Needham-Schroeder protocol [25,29]. Catherine Meadows' work, particularly her work on the NRL protocol analyzer [26] and Maude-NPA [11], has played a great role in this success story. She has used formal models, such as Rewriting Logic and Narrowing, to advance the use of formal methods in protocol security verification.

However, much of the use of formal methods does not consider the protocols timing aspects. An exception is Meadows' work on Distance Bounding (DB) Protocols [28,31], which has been an inspiration to our previous work,[10] and her cost-based framework for analyzing DoS attacks [27]. In a sequence of papers [1,

[10]Indeed, it was Cathy that suggested us to investigate DB protocols.

19, 22, 23], we have developed a number of models based on Multiset Rewriting that investigate different timing aspects of protocols.

A key aspect of Meadows' work is her careful and insightful formalization of important aspects of the assumptions and actions of a protocol. In this spirit, here we describe general Timed Multiset Rewriting (MSR) theories of networks, protocols, and intruders, and show how these theories support representation of diverse timing aspects of protocol execution. We illustrate these aspects with examples. In particular, we model the following timing aspects:

- Network and processing delays, important, *e.g.*, in DB protocol specification and verification;
- Protocol timeouts that have specific applications in a variety of protocols;
- Timed Dolev-Yao intruder, which is similar to the standard Dolev-Yao intruder model in that he can create fresh nonces, compose and decompose messages, for which he possesses the decryption key. However, timed intruder is amended with time features in order to make the physical properties of the system relevant. For example, in contrast to the Dolev-Yao intruder, timed intruder is not able to learn messages immediately, instead, he must obey the restrictions imposed by the physical transmission channel used and wait for a message to reach him.

We illustrate these aspects by specifying DB protocols and protocols with timeouts. The specifications presented here are more general than the ones appearing in our previous work by including all the timing aspects described above.

As an added benefit, we specify verification problems for timed protocol and intruder theories, and obtain some complexity results including the PSPACE-completeness of the secrecy problem. This builds on our past work in which we have developed a rich complexity theory for problems formulated in terms of (Timed) MSR [23].

The paper starts with the description of some timing aspects of security protocols in Section 2. In Section 3, we present the Timed MSR of [23]. In Sections 4 and 5 we define timed protocol and intruder theories. Section 6 introduces relevant verification problems with examples. We present the related complexity results in Section 7. Finally, in Section 8 we conclude by discussing related work and pointing to future work.

2 Timing Aspects of Security Protocols

We illustrate timing aspects of protocols with two examples. The first one is on distance bounding protocols and the second is on the use of timeouts.

2.1 Distance-bounding Protocols

Distance-bounding protocols (DB) [4, 14] aim to enhance traditional authentication with additional assurance of users' physical proximity. The goal of a DB protocol is to provide access to some resource only to valid provers that are within a specified distance bound, and, at the same time, reject access to intruders and to provers that are located outside of the distance bound perimeter. By measuring the round trip time of a challenge-response bit exchange, the verifier deduces the upper bound on the distance of a prover.

Attacks on communication protocols, such as relay attacks on DB protocols, can only be analyzed using models with high-resolution timing information representing physical properties of the communication medium. In order to accommodate such requirements, our models for the formalization and verification of timed protocols include *explicit real time* and specific time aspects involving, *e.g.*, comparisons of time variables.

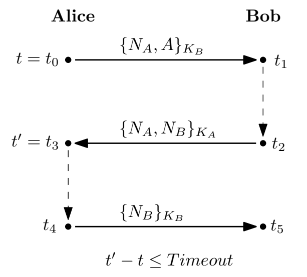
Vulnerabilities of DB protocols, besides cryptographic properties, exploit not only the timing aspects, but also the presence of other honest and dishonest provers, the presence of other verifiers, and colluding intruders, see *e.g.*, [7]. Our model can accommodate such aspects as well. For example, rules of protocol theories (as specified later in Section 4) represent the behavior of verifiers and honest provers. Dishonest provers may be represented through intruder theories (as specified later in Section 5) with specific initial knowledge. Alternatively, theories representing specific behaviors of dishonest provers can be specified, modeling *e.g.*, early responses and guessing. Consequently, verification of DB protocols using our model can reveal various known types of attacks, including in-between-ticks attack [23], distance hijacking [7], among others, including vulnerabilities in multi-protocol environments.

2.2 Timeouts

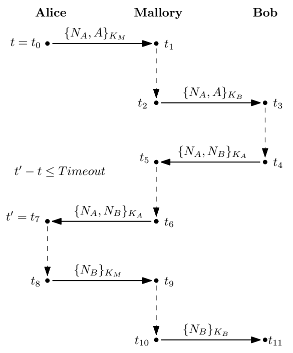
Protocol Session Timeouts become relevant when considering timing aspects, such as, network communication delays. Http/Https protocols use timeouts to limit waiting time in multiple situations: idle connections, client waiting for server responce, server waiting for client to complete a request. The Session Initiation Protocol (used by VOIP and other communication protocols) uses timers to limit the waiting time during different steps of the protocol. For example if the called party is not available the initialization should not ring forever! The ability to reset such timers provides readily available attack surfaces.

Lifetime/time-to-live is another important time related concept. Networking protocols (for example, TLS, Kerberos) often use *tickets* to control access. These tickets typically have a lifetime after which they are no longer valid. Packets traveling through the network (for example TCP/IP) often have a time-to-live to avoid loops and problems delaying delivery.

As an illustration of the use of timeouts, consider the protocol shown in Figure 1a. It is a version of the Needham-Schroeder protocol [29] with timeouts. In particular, Alice starts the communication with Bob as in the original NS, by creating a fresh nonce N_A, at time t_0, reaching Bob at time t_1. Bob answers by creating a fresh nonce N_B and sending it at time t_2, reaching Alice at time t_3. Then Alice responds at time t_4 reaching Bob at time t_5. The difference is that Alice waits for Bob's response for a given period, $Timeout$. If expected message is not received within this period, the protocol session is terminated.



(a) Needham-Schroeder Protocol with Timeout.



(b) Timed Version of Lowe Attack.

Fig. 1: Adding Timeouts to Needham-Schroeder Protocol.

The use of timeouts has implications to an intruder, as shown in the timed version of the Lowe Attack [25] in Figure 1b. In particular, the intruder has to be able to send the response $\{N_A, N_B\}_{K_A}$ to Alice within the period $Timeout$. That is, for that attack to succeed, $t_7 - t_0 \leq Timeout$ has to hold.[11]

For the traditional Dolev-Yao intruder, this is not a problem as he impersonates the network. Thus he can forward messages instantaneously. This may lead to false positives, as it is not physically possible to forward messages instantaneously. We propose in Section 5, therefore, a refinement of the Dolev-Yao intruder which takes timing aspects, such communication and processing delays, into account.

3 Timed Multiset Rewriting

We review Timed Multiset Rewriting of [23] which is the language we use to specify timed protocol and intruder theories. Assume a finite first-order typed alphabet, Σ, with variables, constants, function and predicate symbols. Terms and facts are constructed as usual by applying symbols with correct type [10]. For instance, if P is a predicate of type $\tau_1 \times \tau_2 \times \cdots \times \tau_n \to o$, where o is the type for propositions, and $u_1, \ldots, u_n$ are terms of types $\tau_1, \ldots, \tau_n$, respectively, then $P(u_1, \ldots, u_n)$ is a *fact*. A fact is grounded if it does not contain any variables.

Timestamped facts are used to specify systems that explicitly mention time. Timestamped facts have the form $F@T$, where F is a fact and T is its timestamp, which can be a variable or a *non-negative real number*. There is a special predicate $Time$ with arity zero, used to represent the global time. A configuration is a multiset of ground timestamped facts, $\{Time@t, F_1@t_1, \ldots, F_n@t_n\}$, with a single occurrence of a $Time$ fact.

[11] For simplicity, we ammended only the initiator role, Alice, with a timeout. Since Lowe attack is an attack against both Alice and Bob, the protocol could similary be enhanced with another timeout in the reponder role that would additionally enable Bob to detect that something is wrong.

Actions Actions are multiset rewrite rules and are either time advancement or instantaneous actions. The $Tick$ action, $\mathit{Time}@T \longrightarrow \mathit{Time}@(T+\varepsilon)$, where ε can be instantiated by any positive real number, represents the advancement of time. We also write $Tick_\varepsilon$ when we refer to the $Tick$ rule for a specific ε. Applying the $Tick_\varepsilon$ rule to the configuration $\{\mathit{Time}@t, F_1@t_1, \ldots, F_n@t_n\}$ yields the configuration $\{\mathit{Time}@(t+\varepsilon), F_1@t_1, \ldots, F_n@t_n\}$ where time advances by ε.

The remaining actions are the Instantaneous Actions, which do not affect the global time, but may rewrite the remaining facts. They have the following form:

$$\begin{aligned}&\mathit{Time}@T, W_1@T_1, \ldots, W_k@T_k, F_1@T_1', \ldots, F_n@T_n' \mid \mathcal{C} \longrightarrow\\&\quad \exists \boldsymbol{X}.[\mathit{Time}@T, W_1@T_1, \ldots, W_k@T_k, Q_1@(T+D_1), \ldots, Q_m@(T+D_m)],\end{aligned}$$

where $D_1, \ldots, D_m$ are natural numbers and $\mathcal{C}$ is the guard of the action which is a set of constraints involving the time variables appearing in the pre-condition, *i.e.* the variables $T, T_1, \ldots, T_k, T_1', \ldots, T_n'$. Facts $W_1@T_1, \ldots, W_k@T_k$ are preserved by the rule, while $F_1@T_1', \ldots, F_n@T_n'$ are replaced by $Q_1@(T+D_1), \ldots, Q_m@(T+D_m)$. All free variables appearing in the post-condition shall appear in the pre-condition. Time constraints are of the form:

$$T \geq T' \pm D, \quad T > T' \pm D \quad \text{and} \quad T = T' \pm D$$

where T and T' are time variables, and D is a natural number. In the above rules we omit the time constraints whenever the set $\mathcal{C}$ of time constraints is empty.

An instantaneous rule of the form $\mathcal{P} \mid \mathcal{C} \longrightarrow \exists \boldsymbol{X}.\mathcal{P}'$ can be applied to a configuration $\mathcal{S}$ if there is a subset $\mathcal{S}_0 \subseteq \mathcal{S}$ and a matching substitution θ, such that $\mathcal{S}_0 = \mathcal{P}\theta$ and $\mathcal{C}\theta$ evaluates to true. The resulting configuration from the application of this rule is $(\mathcal{S} \setminus \mathcal{S}_0) \cup ((\mathcal{P}'\sigma)\theta)$, where σ is a substitution that maps the existentially quantified variables $\boldsymbol{X}$ to fresh constants, that is, constants not appearing in $\mathcal{S}$. These fresh values are also called *nonces* in protocol security literature [5, 9].[12] For example, the action

$$\mathit{Time}@T, F_1(X,Y)@T_1 \mid T_1 \geq T+1 \longrightarrow \exists N.[\mathit{Time}@T, F_2(X,Y,N)@(T+3)]$$

can be applied to configuration $\{\mathit{Time}@5.1, F_1(a,b)@7.5\}$ resulting in configuration $\{\mathit{Time}@5.1, F_2(a,b,n_1)@8.1\}$, where the fact $F_1(a,b)@7.5$ is replaced by $F_2(a,b,n_1)@8.1$ with n_1 being a fresh constant. Notice that instantaneous actions do not change the global time. Moreover, the timestamps of the facts that are created by instantaneous actions are in the present or the future.

A trace of timed MSR rules $\mathcal{R}$ from a given initial configuration $\mathcal{S}_0$ is a sequence of configurations $\mathcal{S}_0 \longrightarrow_{r_1} \mathcal{S}_1 \longrightarrow_{r_2} \cdots \longrightarrow_{r_n} \mathcal{S}_n$, such that for all $0 \leq i \leq n-1$, $\mathcal{S}_{i+1}$ is a configuration obtained by applying $r_{i+1} \in \mathcal{R}$ to $\mathcal{S}_i$.

Goal Configurations Among all the possible traces we will be interested in traces that reach some goal. A goal configuration is specified by a *goal* $\mathcal{GS}$ which is a set of pairs $\{ \langle \mathcal{S}_1, \mathcal{C}_1 \rangle, \ldots, \langle \mathcal{S}_n, \mathcal{C}_n \rangle \}$. Each pair $\langle \mathcal{S}_j, \mathcal{C}_j \rangle$ is of the form: $\langle \{F_1@T_1, \ldots, F_p@T_p\}, \mathcal{C}_j \rangle$, where $T_1, \ldots, T_p$ are time variables, $F_1, \ldots, F_p$ are facts and $\mathcal{C}_j$ is a set of time constraints involving only variables $T_1, \ldots, T_p$. A configuration $\mathcal{S}$ is a *goal configuration w.r.t.* $\mathcal{GS}$ if for some $1 \leq i \leq n$, there is a grounding substitution, σ, such that $\mathcal{S}_i\sigma \subseteq \mathcal{S}$ and $\mathcal{C}_i\sigma$ evaluates to true.

[12] Substitution application $(\mathcal{S}\theta)$ is defined as usual [10], *i.e.*, by mapping time variables in $\mathcal{S}$ to non-negative real numbers, nonce names to nonce names (renaming of nonces) and term variables to terms.

For example, the configuration $\{\,Time@10.5,\ F@12.3,\ G(a)@0.1\}$ is a goal configuration w.r.t. the goal $\{\ \langle\ \{Time@T,\ F@T_1\}, \{\ T_1 \geq T\ \}\rangle\ \}$.

Reachablity is one of the main verification problems for MSR systems.

Definition 1. *[Reachability problem] Given a timed MSR $\mathcal{T}$, a goal $\mathcal{GS}$ and an initial configuration $\mathcal{S}_0$, is there a trace, $\mathcal{P}$, that leads from $\mathcal{S}_0$ to a goal configuration?*

Balanced Rules Balanced rules were introduced in [32]. Systems containing only balanced rules represent an important class of systems for which several reachability problems have been shown to be decidable [16,21,23].

A rule is balanced if the number of facts appearing in its pre-condition is the same as the number of facts appearing in its post-condition. An MSR system is balanced if all its rules are balanced.

As described in [16], any unbalanced rule can be made balanced by using so-called *empty facts*. For example, the unbalanced rule: $Time@T, F_1@T_1 \longrightarrow Time@T, F_1@T_1, F_2@T_2$ can be turned into a balanced rule by adding an empty fact to its pre-condition, $Time@T, F_1@T_1, P@T_3 \longrightarrow Time@T, F_1@T_1, F_2@T_2$.

Balanced systems have the following important property: All the configurations in a trace of a balanced system have the same number of facts. Hence, balanced systems are suitable *e.g.*, for modeling scenarios with a fixed amount of memory. As in [16], empty facts represent available free memory slots. In order to model systems and intruders with bounded memory, we will consider empty facts related to the system including agents, servers and the network, D facts, and additionally consider empty facts related to each specific intruder s, $P(s)$.

For some of our complexity results (in Section 7), we will assume an upper-bound on the size of facts. The size, $|F@t|$, of a timed fact $F@t$ is the total number of symbols in F. For example, $|M(a, \{a, b\}_k)@t| = 5$.

4 Timed Protocol Theories

In the traditional "Alice and Bob" protocol notation and specification (such as the one used in Figure 1, Section 2.2) timing aspects of protocols are not formally specified. Neccessary assumptions about time, such as the time requirements for the fulfillment of a protocol session, are not included. For example, in the description of distance-bounding protocols it is only informally described that the verifier remembers the time of sending a challenge bit and the time when receiving the response bit, which are then used to make a decision whether of not to grant access. Moreover, from the traditional protocol description, it is not clear which assumptions about the network are used, such as the transmission medium used by the participants. Furthermore, it is not formally specified which properties does the above protocol ensure, in which conditions, and against which intruders. Security verification should include such specifications when checking whether a system is vulnerable to an attack.

Given a timed MSR model and an initial configuration representing the knowledge of participating agents and intruders, their capabilities and behavior (including protocol rules), we look for a trace representing an attack. For that purpose, a goal configuration will denote that a protocol has suffered an attack.

For protocol and intruder theories that obey the physical laws involving time, it is, in particular, important to consider network delays and processing time. Non-zero processing time can be formalized by adding time constraints to rules, *e.g.*, $Time@T,\ M(m)@T' \mid \{\, T > T' \,\} \longrightarrow Time@T,\ M(m)@T',\ \mathsf{N}_S(m')@T$. Similarly, faithful timing of message transmission for a given network topology between agents can be obtained by adding to transmission rules constraints that involve relevant distances, *e.g.*:

$Time@T, \mathsf{N}_S(A,X)@T' \mid \{\, T \geq T' + D(A,B) \,\} \longrightarrow Time@T, \mathsf{N}_R(B,X)@T$,
where $D(A,B)$ denotes the time required for messages to travel the distance from agent A to agent B.

4.1 Network Theory

We enhance the traditional network models used for protocol execution. A suitable network model used for communication during protocol execution should take care of distances between agents. More specifically, it should not only take care of physical distances between protocol participants, but also represent various available transmission media and the corresponding network distances, *i.e.*, transmission speed, as well as availability of some transmission channel to a particular agent for sending or receiving messages.

We assume that a topology of participating agents, including intruders, representing communicating distances (network distances) between agents, is given. We also assume that agents' and intruder's capabilities of using transmission channels, are given.[13] We model capabilities of agents of sending and receiving messages on some particular transmission media and the corresponding time distances between agents per specific media. Hence, network distances are specified per pair of participants on a specific transmission channel.

We assume that agents do not move. This is also suitable for scenarios where agents may move at a speed that is negligible w.r.t. transmission speed.

Our signature is based on the signature used to model protocols and the Dolev-Yao intruder model in [8,9] and timed Dolev-Yao intruders in [23,30]. In order to provide a finer formalization of the network that supports the timing aspects, we add the following predicate and constant symbols to the signature:

$D(A,B,C)$, natural number denoting the network delay time in communication from agent A to agent B when using transmission media C;

$\mathsf{N}_S(A,C,m)@t$, denoting that message m was sent by agent A on transmission medium C at moment t;

$\mathsf{N}_R(A,C,m)@t$, denoting that message m may be received by agent A on transmission medium C;

$Caps_S(A,C)$, denoting that the agent A is capable of sending messages on transmission medium C;

[13]Instead of such fixed connections of agents to particular channels it is possible to represent agents establishing or dropping connections by additional rules in the model.

$$\begin{array}{l}
\text{NET-1: } Time@T, Cap_S(A,C)@T_1, Cap_R(B,C,1)@T_2, \mathsf{N}_S(A,C,X)@T_3 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mid\ T \geq T_3 + D(A,B,C) \ \longrightarrow \\
\qquad Time@T, Cap_S(A,C)@T_1, Cap_R(B,C,1)@T_2, \mathsf{N}_R(B,C,X)@T \\
\text{NET-2: } Time@T, Cap_S(A,C)@T_1, Cap_R(B,C,2)@T_2, \mathsf{N}_S(A,C,X)@T_3 \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad \mid\ T \geq T_3 + D(A,I,C) \longrightarrow \\
\qquad Time@T, Cap_S(A,C)@T_1, Cap_R(B,C,2)@T_2, \mathsf{N}_S(A,C,X)@T_3, \mathsf{N}_R(B,C,X)@T
\end{array}$$

Fig. 2: Network Theory

$Cap_R(A, C, k)$, denoting that the agent A is capable of receiving messages on transmission medium C, where for $k = 1$ the message is removed from the network, while for $k = 2$ reading does not remove messages from the network.

Network rules are shown in Figure 2. Transmission of messages is encoded as the transformation of N_S facts to N_R facts, *i.e.*, network delivers sent messages according to the receipt capabilities and time distances between participants.

Network theory rules ensure that a message may be received, only after the corresponding message transmission time, so-called "time of flight", has passed. These rules also ensure that agents and intruders only send and receive messages on communication channels they are connected to, *i.e.*, for which they have capabilities of sending or receiving messages. Rule NET-1 models message receipt that removes messages from the network, while NET-2 models non-consumption message receipt, so that the same message X may additionally be received by other agents without re-sending (as in *e.g.*, radio transmission). Which of the two rules is used depends on the nature of the transmission media modeled, which is specified through $Cap_R(X, Y, k)$ facts by having $k = 1$ or $k = 2$.

4.2 Protocol Theories

In the verification of protocols for which time plays a prominent role, such as distance-bounding protocols and cyber-physical systems in general, *explicit real time* is needed for representation of continuity of time in the real physical world.

Our timed MSR model presented in Section 3 is suitable for this purpose. In addition, our model is also suitable for expressing protocols with *timeouts*.

Since the execution and verification of security protocols may be affected by processing time, we add *duration* to the rules specified by DUR function. The arguments and the value of this function are specific to each rule. For example, the length of the plaintext and the key may effect the duration of the encryption. Duration of a rule execution, *i.e.*, DUR function, may be used when suitable, *e.g.*, in verification of attacks that involve the variance of execution time, such as passport traceability attacks in [6]. Rules for which the DUR function is not explicitly mentioned, have zero execution time.

Similarly, protocol states may or may not have timeouts. Once a timeout of a protocol state has passed, the protocol session changes its state.

A general theory of security protocols involving time is specified below. It includes the following predicate symbols:

$E@t$, denoting empty memory slots available for the network and the agents (different from intruders) from the moment t;

$\mathsf{S}_i^A(n, \boldsymbol{X})$, denoting the protocol state predicate of the role A in the session with identifier n;

$\mathsf{T}_i^A(n)@t$, denoting that the protocol state $\mathsf{S}_i^A(n, \boldsymbol{X})$ times out at moment t.

A protocol state S_i associated with a timeout will be accompanied by the corresponding T_i fact, created by the rule leading to that protocol state.

In protocol theories related to traditional security protocols formalized in [9], a protocol execution rule represents an event of a message being received, followed by an immediate message reply. In order to model a variety of protocols, we allow protocol theories where at some protocol state, sending of a message may not be necessarily triggered by a message receipt. Similarly, a receipt of a message at some protocol state, may not be immediately followed by a reply message being sent. As in [9,16] protocols involve a number of roles that can be played by the participants, such as initiator, responder, client or server role.

Definition 2 (Protocol Theory). *A protocol theory $\mathcal{P}$ is specified by a number of roles, $A_1, \ldots, A_m$, and a set of state predicates, $\mathsf{S}_0^{A_i}, \ldots, \mathsf{S}_{n_i}^{A_i}$, and rules of the following form for each role A_i:*

– ***protocol initialization rule:***

$$Time@T, \mathcal{W} \longrightarrow$$
$$\exists S_{id}.[Time@T, \mathsf{S}_0^{A_1}(S_{id}, \boldsymbol{X_1})@(T+t_i), \ldots, \mathsf{S}_0^{A_m}(S_{id}, \boldsymbol{X_m})@(T+t_i), \mathcal{W}'],$$

where S_{id} is a fresh protocol session identification token, $\mathsf{S}_0^{A_i}$ is the initial state of role A_i, $t_i = \textsc{dur}_{INIT}$ is a natural number specifying the time it takes to initialize a protocol session, $\mathcal{W}$ is an arbitrary multiset of facts, $\mathcal{W}' = \mathcal{W} \cup \{\ \mathsf{T}_0^{A_k}(S_{id})@(T+b_k) \mid$ if $\mathsf{S}_0^{A_k}$ timeouts in b_k time units$\}$, and $\boldsymbol{X_i}$ are variables from $\mathcal{W}$;

– ***protocol execution rules** for protocol states $\mathsf{S}_i^{A_k}$ with no timeout:*

$$Time@T, \mathsf{S}_i^{A_k}(S, \boldsymbol{X})@T_1, \mathcal{W}_1,\ \mathcal{W} \ \mid\ T_1 \leq T, \longrightarrow$$
$$\exists \boldsymbol{N}.[Time@T, \mathsf{S}_j^{A_k}(S, \boldsymbol{Y})@(T+t), \mathcal{W}_2, \mathcal{W}]$$

and for protocol states $\mathsf{S}_i^{A_k}$ with associated timeout:

$$Time@T, \mathsf{S}_i^{A_k}(S, \boldsymbol{X})@T_1, \mathsf{T}_i^{A_k}(S)@T_2, \mathcal{W}_1,\ \mathcal{W} \ \mid\ T_1 \leq T, T_2 \geq T \longrightarrow$$
$$\exists \boldsymbol{N}.[Time@T, \mathsf{S}_j^{A_k}(S, \boldsymbol{Y})@(T+t), \mathcal{W}_2, \mathcal{W}],$$

where for $i, j \in \{0, \ldots, k\}$[14] $t = \textsc{dur}_{i,j}(X)$ is a natural number specifying the processing time needed when moving from protocol state $S_i^{A_k}$ to protocol state $S_j^{A_k}$, $\boldsymbol{N}$ are fresh values, $\boldsymbol{X}$ and $\boldsymbol{Y}$ are variables, where variables in $\boldsymbol{Y}$ either appear in facts on the left side of the rule or are freshly generated variables from $\boldsymbol{N}$, and $\mathcal{W}, \mathcal{W}_1, \mathcal{W}_2$ are arbitrary multisets of facts, possibly containing facts N_S or N_R denoting messages being sent and received, where

[14] In our generalization of protocol theories we might omit the condition $i \leq j$ that was the condition in [9] forcing that protocols proceed in execution.

in particular $\mathcal{W}_2$ contains the fact $\mathsf{T}_j^{A_k}(S)@(T+b_j)$ if the protocol state $\mathsf{S}_j^{A_k}$ has the associated timeout (set to expire at moment $T+b_j$);

– ***protocol timeout rules*** *for a protocol state $\mathsf{S}_j^{A_k}$ with associated timeout:*

$$Time@T, \mathsf{S}_j^{A_k}(S, \boldsymbol{X})@T_1, \mathsf{T}_j^{A_k}(S)@T_2 \mid T_2 = T \longrightarrow$$
$$Time@T, \mathsf{S}_i^{A_k}(S, \boldsymbol{X})@(T+t),$$

where $\mathsf{S}_i^{A_k}$ is a protocol state of the role A_k with no associated timeout, or

$$Time@T, \mathsf{S}_j^{A_k}(S, \boldsymbol{X})@T_1, \mathsf{T}_j^{A_k}(S)@T_2 \mid T_2 = T \longrightarrow$$
$$Time@T, \mathsf{S}_i^{A_k}(S, \boldsymbol{X})@(T+t), \mathsf{T}_i^{A_k}(S)@(T+t),$$

where $\mathsf{S}_i^{A_k}$ is a protocol state of the role A_k associated with a timeout, and in both cases $t = \textsc{dur}_{i,j}^{timeout}$ is a natural number specifying transition from state $\mathsf{S}_j^{A_k}$ to state $\mathsf{S}_i^{A_k}$ due to a timeout;

– ***protocol finalization rule****:*

$$Time@T, \mathsf{S}_{n_k}^{A_k}(\boldsymbol{X})@T_1 \mid T_1 \leq T \longrightarrow Time@T,$$

where n_k is natural number specifying the final protocol state $\mathsf{S}_{n_k}^{A_k}$ or role A_k.

All rules of a protocol session are initialized with the same session identifier. Protocol execution rules involve execution time and may also relate to a timeout and network communication. Timeout rules force protocol state change once a timeout has passed, which may result, *e.g.*, in a retry or session termination. Finished sessions are removed by the finalization rule.

For our complexity results we will consider balanced versions of protocol theories that are obtained by adding empty facts on left or right side of the rule, where needed. Such empty facts have timestamps denoting availability or execution time. For example, the balanced version of the protocol finalization rule: $Time@T, \mathsf{S}_{n_k}^{A_k}(\boldsymbol{X})@T_1 \mid T_1 \geq T \longrightarrow Time@T, E@(T+t)$, involves the value t given by the $\textsc{dur}_{FIN}$ function denoting duration of the finalization rule. Such empty memory slots, $E@(T+t)$, are available only when the global time reaches moment $T+t$.

5 Timed Intruder Models

The standard Dolev-Yao intruder (DY) [8] is able to intercept and send messages anywhere at anytime, appearing, hence, faster than the speed of light. For the verification of timed protocols we, therefore, introduce a more adequate, less powerful intruder theories. Our timed intruders still share the capabilities of the standard DY intruder related to composition and decomposition of messages, including encryption and generation of nonces, but in doing so, they respect the physical laws related to time. As in [16], we will also consider intruders with bounded memory.

In order to model the presence of multiple intruders, we associate an identification id to each of the intruders. This id is used to model the knowledge and the memory of a particular intruder through facts $M(id, X)$ and $P(id)$ where:

$M(id, x)@t$ denotes that term x is known to intruder id from the moment t;

I/O Rules:

REC: $\mathit{Time}@T, \mathsf{N}_R(I,C,X)@T_1, P(I)@T_2 \mid T_2 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,X)@(T+t), E@(T+t),$ where $\textsc{dur}_{REC}(X,I)=t$

SND: $\mathit{Time}@T, M(I,X)@T_1, E@T_2 \mid T_1 \le T, T_2 \le T \longrightarrow$
$\quad \mathit{Time}@T, \mathsf{N}_S(I,C,X)@(T+t), P(I)@(T+t),$ where $\textsc{dur}_{SND}(X,I)=t$

Message Composition and Decomposition Rules:

COMP: $\mathit{Time}@T, M(I,X)@T_1, M(I,Y)@T_2 \mid T_1 \le T, T_2 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,\langle X,Y\rangle)@(T+t), P(I)@(T+t'),$ where $\textsc{dur}_{COMP}(X,Y,I)=\langle t,t'\rangle$

DCMP: $\mathit{Time}@T, M(I,\langle X,Y\rangle)@T_1, P(I)@T_2 \mid T_1 \le T, T_2 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,X)@(T+t), M(I,Y)@(T+t),$ where $\textsc{dur}_{DCMP}(\langle X,Y\rangle,I)=t$

USE: $\mathit{Time}@T, M(I,X)@T_1, P(I)@T_2 \mid T_1 \le T, T_2 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,X)@T_1, M(I,X)@(T+t),$ where $\textsc{dur}_{USE}(X,I)=t$

ENC: $\mathit{Time}@T, M(I,K)@T_1, M(I,X)@T_2, P(I)@T_3 \mid T_1 \le T, T_2 \le T, T_3 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,K)@T_1, M(I,X)@T_2, M(I,\{X\}_K)@(T+t),$
where $\textsc{dur}_{ENC}(K,X,I)=t$

DEC: $\mathit{Time}@T, M(I,K^{-1})@T_1, M(I,\{X\}_K)@T_2, P(I)@T_3 \mid T_1 \le T, T_2 \le T, T_3 \le T \longrightarrow$
$\quad \mathit{Time}@T, M(I,K^{-1})@T_1, M(I,\{X\}_K)@T_2, M(I,X)@(T+t),$
where $\textsc{dur}_{DEC}(K^{-1},\{X\}_K,I)=t$

GEN: $\mathit{Time}@T, P(I)@T_1 \mid T_1 \le T \longrightarrow \exists N.\mathit{Time}@T, M(I,N)@(T+t),$
where $\textsc{dur}_{GEN}(I)=t$

Memory Maintenance Rule:

DELM: $\mathit{Time}@T, M(I,X)@T_1 \mid T_1 \le T \longrightarrow \mathit{Time}@T, P@(T+t),$
where $\textsc{dur}_{DEL}(X,I)=t$

Fig. 3: Bounded Memory Timed DY Intruder Theory $\mathcal{I}$

$P(id)@t$ denotes that a memory slot (empty fact) is available to the bounded memory intruder id from the moment t.

As already mentioned, in order to model processing time, each of the intruder rules has an associated time cost. This is specified by the associated function $\textsc{dur}$, returning the time needed to carry out the action (as detailed below). This allows us to model the standard message processing time where *e.g.*, encryption takes much more time than composition of a pair of messages.

5.1 Bounded Memory Timed Dolev-Yao Intruder

Intruder rules of a *Bounded Memory Timed DY Intruder* are balanced and contain empty facts representing memory available to the intruder, see Figure 3.

The general timed DY intruder theory with unbounded memory is obtained by dropping the empty facts and the memory management rule from the theory in Figure 3.

At the time T intruder I can access only known terms, $M(I,X)@T'$, and empty memory slots, $P(I)@T'$, only if $T' \le T$. All empty facts, $P(I)@T'$, that appear on the left side of a rule have the associated time constraint, $T' \leq T$, to ensure that a memory slot is available at current time, $Time@T$.

Deleting facts from the memory (denoted by the DELM rule) may also take time. The COMP rule has two associated time constants through DUR function, one denoting the time it takes to produce a pair of messages, and the other denoting the time it takes to make an empty fact available. The REC and SND rules are related to receiving and sending messages on transmission media that is available to some intruder. Notice that intruders obey physical laws related to message delivery and transmission media availability, which are enforced through network theory, given in Figure 2, for all agents, including intruders. Notice as well that send and receive rules maintain the total memory of intruder and the total memory of the system, by consuming or creating P and E facts.

An adversary can also jam a channel by sending a large number of messages, exhausting the system's network by consuming E facts through the SND rule. For the representation of specific channel capacities, special empty facts representing the network bandwidth could be added and associated to each channel.

For specific scenarios, other intruder capabilities may be relevant, such as intruder capabilities of message manipulation on the wireless channels, modeled, *e.g.*, in [7]. This includes overshadowing parts of a message, as well as flipping some bits of a message. Such capabilities, we believe, could be formalized in our model as well, by adding the Xor function to the signature and by adding the corresponding intruder rules.

6 Verification Problems

Reachability and the related problems for MSR are undecidable in general [21]. By imposing some restrictions, such as using only balanced rules and bounding the size of facts, these problems become decidable, even in timed models with fresh values [17,23]. Balanced systems used for protocol verification, as the ones in [16,23], implicitly bound the number of protocol sessions that can be executed concurrently. However, the number of sessions in a trace is unbounded.

Various problems can be considered in the verification of security protocols. Here we state some of them.

Definition 3 (Secrecy Problem). *Given a protocol theory $\mathcal{P}$, network theory $\mathcal{N}$, intruder theories $\mathcal{I}_1, \ldots \mathcal{I}_k$ and an initial configuration $\mathcal{S}_0$ denoting the initial protocol setting including key distribution, communication capabilities, network distances and a constant s known only to some agent, the* secrecy problem of a protocol theory $\mathcal{P}$ *is the problem of determining whether or not a configuration containing the fact $M(I, s)$, for some intruder identifier I, is reachable from $\mathcal{S}_0$ using rules in $\mathcal{N}, \mathcal{I}_1, \ldots, \mathcal{I}_k$ and $\mathcal{P}$.*

In other words, the secrecy problem is the problem of determining whether or not an intruder can learn the secret s, initially known to some honest agent.

A more general version of the secrecy problem involving several different protocol theories, $\mathcal{P}_1, \ldots, \mathcal{P}_k$, suitable for verification of multi-protocol environments, is analogously defined.

Next, we define verification problems related to DB protocols.

Definition 4 (False Acceptance Problem). *Given a DB protocol theory $\mathcal{P}$ with a distance bound R, network theory $\mathcal{N}$, intruder theories $\mathcal{I}_1, \ldots \mathcal{I}_k$ and an initial configuration $\mathcal{S}_0$ denoting the initial protocol setting including key distribution, communication capabilities etc., the* false acceptance problem *is the problem of determining whether or not a configuration denoting that a verifier has granted access to an intruder or to a prover that is outside the perimeter R, is reachable from $\mathcal{S}_0$ using rules in $\mathcal{N}, \mathcal{I}_1, \ldots, \mathcal{I}_k$ and $\mathcal{P}$.*

A dual problem related to decision errors for DB protocols, is the following.

Definition 5 (False Rejection Problem). *Given a DB protocol theory $\mathcal{P}$ with a distance bound R, network theory $\mathcal{N}$, intruder theories $\mathcal{I}_1, \ldots \mathcal{I}_k$ and an initial configuration $\mathcal{S}_0$ denoting the initial protocol setting including key distribution, communication capabilities etc., the* false rejection problem *is the problem of determining whether or not a configuration denoting that a verifier has denied access to an honest prover that is within the perimeter R, is reachable from $\mathcal{S}_0$ using rules in $\mathcal{N}, \mathcal{I}_1, \ldots, \mathcal{I}_k$ and $\mathcal{P}$.*

By including several DB protocol theories, multi-protocol environments, as in [7], can be verified. In our recent work [1] on DB protocols we have also investigated Attack Detection Problem. We believe other problems such as Denial of Service, as well as other classes of problems involving *e.g.*, privacy and traceability could also be formulated in our model. We leave this for future work.

6.1 Example: The Hancke-Khun Protocol Theory

We now illustrate the expressiveness of timed protocol theories introduced in Section 4 by formalizing the Hancke-Khun (HK) distance-bounding protocol [14].

The HK protocol, shown in Figure 4, aims to ensure that the prover, P, is in the vicinity of the verifier, V. It is assumed that the prover and the verifier share a long-term secret key, K, and a public hash function, h.

In the initial phase of the protocol the verifier and the prover generate nonces N_V and N_P which are used to calculate a sequence of $2n$ bits using K and h: $\boldsymbol{h} = h(K, N_V, N_P) = R_1^0, \ldots, R_n^0 || R_1^1, \ldots, R_n^1,\ R_i^j \in \{0,1\}$. Let $\boldsymbol{s} = \langle K, h, N_V, N_P \rangle$ denote data that, together with $\boldsymbol{h}$, is known to both participants after the initial phase of a particular protocol session.

The setup phase of HK protocol is followed by a series of n single-bit exchanges, defined by the following procedure: To a random challenge bit C_i sent by the verifier in the ith round, the prover instantly replies with either R_i^0, in case $C_i = 0$, or R_i^1, in case $C_i = 1$. We formalize verifier's random bit generation of challenge bits using nonce generation and a function b that returns a bit, *i.e.*, $b(x) \in \{0,1\}$. Comparison of received responses with the correct bits in $\boldsymbol{h}$, precalculated in the initial phase of the protocol, is obtained using a function r that returns the bit $R_i^{C_i}$ based on bit C_i and $\boldsymbol{h}$ for ith round, as per H-K protocol specification, *i.e.*, $r(i, \boldsymbol{h}, x) = \begin{cases} h_i, & x = 0, \\ h_{n+i}, & x = 1. \end{cases}$.

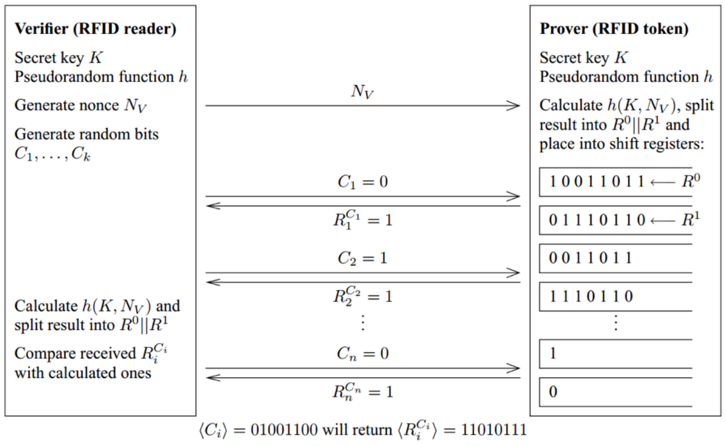

Verifier (RFID reader)
Secret key K
Pseudorandom function h
Generate nonce N_V
Generate random bits $C_1, \ldots, C_k$
Calculate $h(K, N_V)$ and split result into $R^0||R^1$
Compare received $R_i^{C_i}$ with calculated ones
N_V
$C_1 = 0$
$R_1^{C_1} = 1$
$C_2 = 1$
$R_2^{C_2} = 1$
$C_n = 0$
$R_n^{C_n} = 1$
Prover (RFID token)
Secret key K
Pseudorandom function h
Calculate $h(K, N_V)$, split result into $R^0||R^1$ and place into shift registers:
1 0 0 1 1 0 1 1 ⟵ R^0
0 1 1 1 0 1 1 0 ⟵ R^1
0 0 1 1 0 1 1
1 1 1 0 1 1 0
1
0
$\langle C_i \rangle = 01001100$ will return $\langle R_i^{C_i} \rangle = 11010111$

Fig. 4: The Hancke-Kuhn Protocol (taken from [14])

For each round, the verifier marks the time when a challenge bit is sent, and the time the response is received. In the last phase of the protocol, the verifier computes his distance from the prover and checks that the responses are correct.

The verifier grants access to the prover if all time tests for bit exchanges are successful, i.e., do not exceed the predefined distance bound, R, and if all n bits are correctly exchanged. Keeping in mind potential errors, due to e.g., noise, the verifier's decision can be parametrised so that access is granted if the time-test is satisfied in a number of rounds, k_1 out of n, e.g., in a simple majority of rounds, and if a certain number of response bits, k_2 out of n, are correct.

For illustration purposes we only formalize the bit exchange phase of the HK protocol with n challenge-response rounds, see Figure 5. The initial phase of the HK protocol could be similarly formalized. Here, we assume that the initial phase of HK protocol session S has already been completed, denoted by the facts $\mathsf{S}_0^V(S, \boldsymbol{s}, \boldsymbol{h}, 1)$ and $\mathsf{S}_0^P(S, \boldsymbol{s}, \boldsymbol{h}, 1)$, representing the initial protocol states for the verifier and the prover roles, respectively. Completion of n rounds of bit exchanges is denoted by the fact $\mathsf{S}_4^V(S, \boldsymbol{s}, \boldsymbol{h})$.

Besides agents' capabilities of using transmission media, keys etc., the initial configuration additionally includes the following auxiliary facts: $Bits(S, 0)@0$ denoting the number of rounds with correct bit responses, $Test(S, 0)$ denoting the number of rounds successfully passing the time-test.

Since this distance measuring phase of the protocol is technically performed with negligible processing time related to reading and responding with bits, we set the related processing time to zero.

However, in order to model actual verifiers that are usually not very powerful processors operating at some clock rate, the formalization distinguishes between the actual time of sending challenge bits or receiving response bits and the recorded time. This is accomplished using time constraints of the form $T > T_1$, where T is the global time and T_1 the actual time of sending or receiving the bit. Alternatively, function DUR could be used for specific time delays.

Verifier role

Send challenge bit for round j :

$Time@T, \mathsf{S}_0^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j)@T_1, E@T_2, E@T_3 \mid \{\ T \geq T_2, T \geq T_3\ \} \longrightarrow$
$\quad \exists x.Time@T, \mathsf{S}_1^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{pending}, b(x), j)@T, \mathsf{N}_S(A, C, b(x))@T, Start(S, j)@T$

Mark the time of sending the challenge bit in round j :

$Time@T, \mathsf{S}_1^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{pending}, B_j, j)@T_1, E@T_2 \mid \{\ T > T_1, T \geq T_2\ \} \longrightarrow$
$\quad Time@T, \mathsf{S}_1^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{start}, B_j, j)@T, Start_V(S, j)@T$

Receive the response bit in round j :

$Time@T, \mathsf{S}_1^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{start}, B_j, j)@T_1, \mathsf{N}_R(A, C, X)@T_2 \longrightarrow$
$\quad Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{pending}, B_j, j, X)@T_1, Stop(S, j)@T$

Mark the time of receiving the response bit in round j :

$Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{pending}, B_j, j, X)@T_1,\ E@T_2 \mid \{\ T > T_1\ \} \longrightarrow$
$\quad Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{stop}, B_j, j, X)@T_1, Stop_V(S, j)@T$

Check the round trip time for round j :

$Time@T, Start_V(S, j)@T_1, Stop_V(S, j)@T_2, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{stop}, B_j, j, X)@T_3,$
$\quad Test(S, m)@T_4 \mid \{\ T_2 - T_1 \leq 2R\} \longrightarrow$
$\quad Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{bit}, B_j, j, X)@T, TimeCheck(S, j, \text{ok})@T, Test(S, m + 1)@T, E@T$

$Time@T, Start_V(S, j)@T_1, Stop_V(S, j)@T_2, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{stop}, B_j, j, X)@T_3,$
$\quad Test(S, m)@T_4 \mid \{\ T_2 - T_1 > 2R\} \longrightarrow$
$\quad Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{bit}, B_j, j, X)@T, TimeCheck(S, j, \text{not-ok})@T, Test(S, m)@T_4, E@T$

Check the bit correctness in round j :

$Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{bit}, B_j, j, r(j, \boldsymbol{h}, B_j))@T_1, Bit(m)@T_2, E@T_3 \mid \{T_3 \leq T\ \}$
$\quad \longrightarrow\ Time@T, \mathsf{S}_3^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T, Bit(m + 1)@T, BitCheck(S, j, \text{ok})@T$

$Time@T, \mathsf{S}_2^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, \text{bit}, B_j, j, y \neq r(j, \boldsymbol{h}, B_j))@T_1, Bit(m)@T_2, E@T_3 \mid \{T_3 \leq T\ \}$
$\quad \longrightarrow\ Time@T, \mathsf{S}_3^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j)@T, Bit(m)@T_2, BitCheck(S, j, \text{not-ok})@T$

Starting a new round or finishing the last round :

$Time@T, \mathsf{S}_3^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j \neq n)@T_1 \ \longrightarrow\ Time@T, \mathsf{S}_0^V(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j + 1)@T$

$Time@T, \mathsf{S}_3^V(S, A, \boldsymbol{s}, \boldsymbol{h}, n)@T_1 \ \longrightarrow\ Time@T, \mathsf{S}_4^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T$

Allowing or rejecting the access :

$Time@T, \mathsf{S}_4^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T_1, Test(S, X \geq k_1)@T_2, Bit(S, Y \geq k_2)@T_3 \ \longrightarrow$
$\quad Time@T, \mathsf{S}_5^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T, Decision(S, \text{ok})@T, E@T$

$Time@T, \mathsf{S}_4^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T_1, Test(S, X < k_1)@T_2, Bit(S, Y < k_2)@T_3 \ \longrightarrow$
$\quad Time@T, \mathsf{S}_5^V(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T, Decision(S, \text{reject})@T, E@T$

Prover role

Responding to a challenge bit in round j :

$Time@T, \mathsf{S}_0^P(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j \neq n)@T_1, \mathsf{N}_R(A, C, X)@T_2 \longrightarrow$
$\quad Time@T, \mathsf{S}_0(S, A, C, \boldsymbol{s}, \boldsymbol{h}, j + 1)@T, \mathsf{N}_S(A, C, r(i, \boldsymbol{h}, X))@T$

Responding to a challenge bit in the last round :

$Time@T, \mathsf{S}_0^P(S, A, C, \boldsymbol{s}, \boldsymbol{h}, n)@T_1, \mathsf{N}_R(A, C, X)@T_2 \longrightarrow$
$\quad Time@T, \mathsf{S}_1(S, A, C, \boldsymbol{s}, \boldsymbol{h})@T, \mathsf{N}_S(A, C, r(i, \boldsymbol{h}, X))@T$

Receiving the decision :

$Time@T, Decision(S, X)@T_1, \mathsf{S}_1^P(S, \boldsymbol{Y})@T_2 \longrightarrow Time@T, Decision(S, X)@T_1, \mathsf{S}_2^P(S, \boldsymbol{Y})@T$

Fig. 5: Protocol Theories for bit exchange phase of Hancke-Kuhn protocol with n rounds

Checking whether a sufficient number of rounds have passed the time-test and the bit correctness test is part of the final phase of HK protocol. In our formalization we have included these rules in each round, but with no time cost.

For the HK protocol specification given in Figure 5, with the the protocol distance bound R, the false acceptance problem representing an attack-in-between ticks is specified as a reachability problem with the following goal configuration:

$$\{\ \mathit{Start}(S, X_1)@T_1^1, \mathit{Stop}(S, X_1)@T_2^1, \ldots, \mathit{Start}(S, X_{k_1})@T_1^{k_1}, \mathit{Stop}\ (S, X_{k_1})@T_2^{k_1}, \mathit{Decision}(S, \mathrm{ok})@T,\ \}\ \mid\ \{\ T_2^1 - T_1^1 > R, \ldots T_2^{k_1} - T_1^{k_1} > R\ \},$$

for some protocol session S and k_1 rounds X_i. Similarly, a more general false acceptance can be formalized with the goal configuration:

$$\mathsf{S}_5^V(S, A, C, \boldsymbol{X})@T, \mathsf{S}_2^P(S, B, C, \boldsymbol{Y})@T_1, \mathit{Decision}(S, \mathrm{ok})@T_2,$$

where $D(A, B, C) > R$. This goal denotes a false positive of the time test, *i.e.*, the verifier allows access to a prover that is outside the perimeter R.

False rejction can similarly be represented with the following goal:

$$\mathsf{S}_5^V(S, A, C, \boldsymbol{X})@T, \mathsf{S}_2^P(S, B, C, \boldsymbol{Y})@T_1, \mathit{Decision}(S, \mathrm{reject})@T_2,$$

where $D(A, B, C) \leq R$, for the protocol distance bound R.

Guessing ahead attacks could also be captured by checking whether response bits are received before the necessary traversal time, as specified by the goal:

$$\mathit{Decision}(S, \mathrm{ok})@T, \mathsf{S}_5^V(S, A, C, \boldsymbol{X})@T, \mathsf{S}_2^P(S, B, C, \boldsymbol{Y})@T_1,$$
$$\mathit{Start}(S, i)@T_2, \mathit{Stop}\ (S, i)@T_3 \mid T_3 - T_2 < 2D(A, B, C).$$

This indicates that the response bit has been sent in advance, before the receipt of the challenge bit, representing, hence, guessing in advance, *i.e.*, involvement of a dishonest prover or an intruder.

6.2 Example: The Needham-Schroeder Protocol with Timeouts

We specify the Needham-Schroeder (NS) protocol with timeouts detailed in Section 2.2. Initiator role A has the associated timeout. Only if the expected reply message is received within the timeout time bound, the final protocol message is sent. Otherwise, the session ends.

A balanced timed protocol theory of NS with timeouts is given in Figure 6. The protocol state S_0^A has the associated timeout. Here, $*$ denotes a dummy constant, d_i are constants denoting action duration and b_0 is the constant denoting the timeout. For simplicity, we use public keys to denote names of agents, where *Agent* predicate is used to specify public keys, while key pairs of public and private keys that belong to an honest participant are denoted by *Guy* facts.

Protocol security is considered in the usual sense, *i.e.*, if the "accepted" nonces N_A and N_B are never revealed to anybody else except Alice and Bob executing the protocol. The protocol is still vulnerable to the timed version of Lowe attack [25], see Figure 1b, but a well-chosen timeout may enhance protocol security.

For illustration, let $d_0 = d_1 = d_2 = d_3 = d_4 = 1$, *i.e.*, all actions take one time unit to be executed. Let $b_0 = 10$, *i.e.*, state S_0^A timeouts after 10 time units. Let $D(k_A, k_B, c) = D(k_B, k_A, c) = 3$. In this setting, execution of rules ROLES, A1, B1 and network rules takes at least (1+1+3+1+3=9) 9 time units, *i.e.*, Alice can expect to receive the reply within the set timeout and proceed protocol execution with rule A2.

$$
\begin{array}{l}
\text{ROLES}: \mathit{Time}@T, \mathit{Guy}(K_e,K_d)@T_1, \mathit{Guy}(K'_e,K'_d)@T_2, E@T_3, E@T_4 \mid T_3 \le T, T_4 \le T \\
\qquad\qquad \rightarrow\ \exists X.\mathit{Time}@T, \mathit{Guy}(K_e,K_d)@T_1, \mathit{Guy}(K'_e,K'_d)@T_2, \\
\qquad\qquad\qquad \mathsf{S}_0^A(X,K_e)@(T+d_0), \mathsf{T}_0^A(X)@(T+b_0), \mathsf{S}_0^B(X,K'_e)@(T+d_0) \\
\text{FINA}: \quad \mathit{Time}@T, \mathsf{S}_2^A(\boldsymbol{X})@T_1 \ \rightarrow\ \mathit{Time}@T, E@T \\
\text{FINB}: \quad \mathit{Time}@T, \mathsf{S}_2^B(\boldsymbol{X})@T_1 \rightarrow\ \mathit{Time}@T, E@T \\
\text{A1}: \mathit{Time}@T, \mathsf{S}_0^A(S,K_e)@T_1, \mathsf{T}_0^A(S)@T_2, \mathit{Agent}(K'_e)@T_3 \mid T_2 \ge T, T_1 \le T \\
\quad \rightarrow\ \exists X.\mathit{Time}@T, \mathsf{S}_1^A(S,K_e,K'_e,X)@(T+d_1), \mathit{Agent}(K'_e)@T_3, \\
\qquad \mathsf{N}_S(K_e,C,\mathit{enc}(K'_e,\langle X,K_e\rangle))@(T@d_1) \\
\text{A2}: \mathit{Time}@T, \mathsf{S}_1^A(S,K_e,K'_e,X)@T_1, \mathsf{N}_R(K_e,C,\mathit{enc}(K_e,\langle X,Y\rangle))@T_2 \mid T_1 \le T \\
\quad \rightarrow\ \mathit{Time}@T, \mathsf{S}_2^A(S,K_e,K'_e,X,Y)@(T+d_2), \mathsf{N}_S(K_e,C',\mathit{enc}(K'_e,Y))@(T+d_2) \\
\text{AT}: \mathit{Time}@T, \mathsf{S}_0^A(S,X)@T_1, \mathsf{T}_0^A(S)@T_2 \ \mid\ T_2 = T \ \longrightarrow\ \ \mathit{Time}@T, \mathsf{S}_2^A(S,X,*,*,*)@T_1 \\
\text{B1}: \mathit{Time}@T, \mathsf{S}_0^B(S,K_e)@T_1, \mathit{Agent}(K'_e)@T_2, \mathsf{N}_R(K_e,C,\mathit{enc}(K_e,\langle X,K'_e\rangle))@T_3 \mid T_1 \le T \\
\quad \rightarrow \exists Y.\mathit{Time}@T, \mathsf{S}_1^B(S,K_e,K'_e,X,Y)@(T+d_3), \mathit{Agent}(K'_e)@T_2, \\
\quad \mathsf{N}_S(K_e,C',\mathit{enc}(K'_e,\langle X,Y\rangle))@(T+d_3) \\
\text{B2}: \mathit{Time}@T, \mathsf{S}_1^B(S,K_e,K'_e,X,Y)@T_1, \mathsf{N}_R(K_e,C,\mathit{enc}(K_e,Y))@T_2 \mid T_1 \le T \\
\quad \rightarrow\ \mathit{Time}@T, \mathsf{S}_2^B(S,K_e,K'_e,X,Y)@(T+d_4), E@T
\end{array}
$$

Fig. 6: Timed protocol theory of Needham-Schroeder Protocol with Timeouts.

Consider now the setting with Mallory positioned optimally, inbetween Alice and Bob, as illustrated in Figure 1b, with $D(k_A, k_M, c) = D(k_M, k_A, c) = 2$ and $D(k_B, k_M, c) = D(k_M, k_B, c) = 1$. Let Mallory intruder rules also have associated time cost of one time unit, for simplicity. Mallory will need to intercept, decrypt, encrypt and send messages which will take some additional time. Now, protocol, intruder and network execution of rules from ROLES up to A2 rule take at least 15 time units (Mallory has to use the sequence of REC, DEC, ENC, SND rules, and later REC and SND rules), by which time the protocol session timeouts and, hence, Lowe type attack fails. In case the timeout is set to a high enough value, *e.g.*, $b_0 = 20$, there is a trace in the model representing the Lowe attack.

7 Complexity Results

Timed MSR theories containing network, protocol and intruder theories defined in Sections 4 and 5 represent a segment of general timed MSR for specification and verification of security protocols. By relying on our previous complexity results for the secrecy problem and for the reachability problem for timed MSR, we obtain the complexity result for the timed version of secrecy problem described in Section 6. We point out that this verification relates to traces with a bounded number of concurrent protocol sessions, but to an unbounded number of protocol sessions in total.

Theorem 1. *The secrecy problem with respect to the memory bounded timed DY intruders, balanced network and protocol theories is PSPACE-complete when assuming a bound on the size of facts.*

Proof. For the upper bound we rely on the PSPACE-completeness of the reachability problem for general MSR with real time [23], since the secrecy problem is an instance of the reachability problem. The rules of the timed MSR contain network, protocol and intruder theories. The goal of the reachability problem is specified as a configuration containing $M(I, s)@T$, for some intruder identifier I, with no time constraints attached. Therefore, it follows from [23] that the secrecy problem is in PSPACE when considering balanced timed network, protocol and intruder theories with a bound on the size of facts.

The lower bound follows from PSPACE-completeness of the secrecy problem for untimed version of bounded memory intruder and balanced MSR protocol theories [16]. It can be encoded as timed secrecy simply by adding timestamps to facts, by adding time constraints to the rules of protocol and intruder theories, as per Definition 2 and Figure 3, and by considering some arbitrary network topology and a single transmission channel, accessible to all agents and intruders both for sending and receiving messages. In particular, the protocol states of timed protocol theories have no timeouts attached. Exact values of timestamps, duration of rules, just like the message traversal time (specified by the topology) have no impact to the encoding. Indeed, the goal involves the secret being discovered by an intruder, taking any amount of time, as the goal involves no related time constraints, and all constraints attached to rules of network, intruder and protocol theories (since no timeouts are present) are of the form $T' \leq T$, *i.e.*, require only advancement of global time T, which is always applicable.

False acceptance and false rejection problems could also be formalized as reachability problems, as in Section 6.1, hence the PSPACE membership of these problems can also be deduced.

Theorem 2. *The false acceptance and the false rejection problems with respect to the memory bounded timed DY intruders, balanced network and distance-bounding protocol theories is in PSPACE when the size of facts is bounded.*

Similarly to the bounded-time problems introduced in [20], we could consider bounded time version of the secrecy problem, *e.g.*, by bounding the total time in a trace or bounding the total number of protocol sessions. We expect that such restrictions would improve the complexity of the problem. We leave this investigation for future work.

8 Conclusions and Related Work

This paper builds on [1,19,22,23] and introduces a uniform and extensible framework for expressing a wide range of timing properties of protocols enabling the investigation of the complexity of different verification problems. Thus, this work is complementary to the related works that focus on more limited languages in order to automate analyses.

The first full-scale formal representation and analysis of a distance bounding protocol is the work of Meadows and collaborators [28] formalizing distance bounding protocols in Protocol Derivation Logic (PDL). This work provided the basis for practical improvements, new insights, and inspirations for other researchers including ourselves. Like our work that is founded on an existing general model (Timed MSR), the formalization started with an existing formal logic, PDL.

The paper [30] introduces a timed protocol language and addresses the issue of timed intruder models, showing that one DY intruder per honest player is sufficient. This formal system is implemented in Maude to automate analysis. This work built on earlier formalizations in Timed MSR [18,22] and in turn has suggested new modeling challenges addressed in the present paper.

A number of other frameworks have been developed for the verification of timing properties of systems. Early examples include [3,12,13,15]. Basin et.al [2] and Cramers et.al [7] present a formalism for representing and analyzing cyber-physical security protocols that is implemented in Isabel/HOL. They model physical properties of communicaiton, location, and time. Similar to our approach both honest players and intruders are subject to physical constraints. A benefit of formalization in our Timed MSR is the abiltiy to leverage a variety of complexity results developed for different fragments as illustrated in the previous section.

Verification in this paper assumes that a concrete topology of agents and intruders is specified. We believe it may be possible to consider verification of general topologies by combining our models with SMT solvers, similarly to [30].

For a close formalization of DB protcols probabilities involving various guessing strategies as given in [1], we intend to investigate ways of extending our models with probabilities. One of the ways of such probabilistic extensions of our models might involve branching actions introduced in [24].

Acknowledgments: We thank Cathy for her inspiring work, insightful and motivating discussions and for her friendship.

References

1. M. A. Alturki, M. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Statistical model checking of distance fraud attacks on the Hancke-Kuhn family of protocols. In *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*, pages 60–71. ACM, 2018.
2. D. A. Basin, S. Capkun, P. Schaller, and B. Schmidt. Formal reasoning about physical properties of security protocols. *ACM Trans. Inf. Syst. Secur.*, 14(2):16, 2011.
3. G. Bella and L. C. Paulson. Kerberos version 4: Inductive analysis of the secrecy goals. In *Computer Security - ESORICS 98, 5th European Symposium on Research in Computer Security, Louvain-la-Neuve, Belgium, September 16-18, 1998, Proceedings*, pages 361–375, 1998.
4. S. Brands and D. Chaum. Distance-bounding protocols. In T. Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93: Workshop on the Theory and Application of Cryptographic Techniques Lofthus, Norway, May 23–27, 1993 Proceedings*, pages 344–359, Berlin, Heidelberg, 1994. Springer.

5. I. Cervesato, N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In *CSFW*, pages 55–69, 1999.
6. T. Chothia and V. Smirnov. A traceability attack against e-passports. In *International Conference on Financial Cryptography and Data Security*, pages 20–34. Springer, 2010.
7. C. Cremers, K. B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *2012 IEEE Symposium on Security and Privacy*, pages 113–127, May 2012.
8. D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, 1983.
9. N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004.
10. H. B. Enderton. *A mathematical introduction to logic.* Academic Press, 1972.
11. S. Escobar, C. Meadows, and J. Meseguer. *Maude-NPA: Cryptographic Protocol Analysis Modulo Equational Properties*, pages 1–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
12. N. Evans and S. Schneider. Analysing time dependent security properties in CSP using PVS. In *Computer Security - ESORICS 2000, 6th European Symposium on Research in Computer Security, Toulouse, France, October 4-6, 2000, Proceedings*, pages 222–237, 2000.
13. R. Gorrieri, E. Locatelli, and F. Martinelli. A simple language for real-time cryptographic protocol analysis. In *Proceedings of the 12th European Conference on Programming*, ESOP'03, pages 114–128, Berlin, Heidelberg, 2003. Springer-Verlag.
14. G. P. Hancke and M. G. Kuhn. An RFID distance bounding protocol. In *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*, pages 67–73, Sept 2005.
15. G. Jakubowska and W. Penczek. Modelling and checking timed authentication of security protocols. *Fundam. Inf.*, 79(3-4):363–378, Aug. 2007.
16. M. Kanovich, T. Ban Kirigin, V. Nigam, and A. Scedrov. Bounded memory Dolev-Yao adversaries in collaborative systems. *Inf. Comput.*, 2014.
17. M. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Compliance in real time multiset rewriting models. Available at https://arxiv.org/abs/1811.04826.
18. M. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Discrete vs. dense times in the analysis of cyber-physical security protocols. In *Principles of Security and Trust - 4th International Conference, POST*, pages 259–279, 2015.
19. M. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Can we mitigate the attacks on distance-bounding protocols by using challenge-response rounds repeatedly? In *FCS*, 2016.
20. M. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. Talcott. Timed multiset rewriting and the verification of time-sensitive distributed systems. In *14th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS)*, 2016.
21. M. Kanovich, P. Rowe, and A. Scedrov. Policy compliance in collaborative systems. In *CSF '09: Proceedings of the 2009 22nd IEEE Computer Security Foundations Symposium*, pages 218–233, Washington, DC, USA, 2009. IEEE Computer Society.
22. M. I. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. L. Talcott. Towards timed models for cyber-physical security protocols. Available in Nigam's homepage, 2014.

23. M. I. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, and C. L. Talcott. Time, computational complexity, and probability in the analysis of distance-bounding protocols. *Journal of Computer Security*, 25(6):585–630, 2017.
24. M. I. Kanovich, T. Ban Kirigin, V. Nigam, A. Scedrov, C. L. Talcott, and R. Perovic. A rewriting framework and logic for activities subject to regulations. *Mathematical Structures in Computer Science*, 27(3):332–375, 2017.
25. G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *TACAS*, pages 147–166, 1996.
26. C. Meadows. The nrl protocol analyzer: An overview. *The Journal of Logic Programming*, 26(2):113 – 131, 1996.
27. C. Meadows. A cost-based framework for analysis of denial of service in networks. *J. Comput. Secur.*, 9(1-2):143–164, Jan. 2001.
28. 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*, pages 279–298. 2007.
29. R. M. Needham and M. D. Schroeder. Using encryption for authentication in large networks of computers. *Commun. ACM*, 21(12):993–999, 1978.
30. V. Nigam, C. L. Talcott, and A. A. Urquiza. Towards the automated verification of cyber-physical security protocols: Bounding the number of timed intruders. In *21st European Symposium on Research in Computer Security. Part II*, pages 450–470, 2016.
31. D. Pavlovic and C. Meadows. Bayesian authentication: Quantifying security of the Hancke-Kuhn protocol. *Electronic Notes in Theoretical Computer Science*, 265:97–122, 2010.
32. P. Rowe. *Policy compliance, confidentiality and complexity in collaborative systems.* PhD thesis, University of Pennsylvania, 2009.