Int J Inf Secur (2004) / Digital Object Identifier (DOI) 10.1007/s10207-004-0050-z

On the decidability of cryptographic protocols with open-ended data structures

Ralf Küsters*

Institut für Informatik, Christian-Albrechts-Universität zu Kiel, 24098 Kiel, Germany
e-mail: kuesters@ti.informatik.uni-kiel.de

Published online: 20 October 2004 –

Abstract. Formal analysis of cryptographic protocols has concentrated mainly on protocols with closed-ended data structures, i.e., protocols where the messages exchanged between principals have fixed and finite format. In many protocols, however, the data structures used are open-ended, i.e., messages have an unbounded number of data fields. In this paper, decidability issues for such protocols are studied. We propose a protocol model in which principals are described by transducers, i.e., finite automata with output, and show that in this model security is decidable and PSPACE-hard in presence of the standard Dolev-Yao intruder.

Keywords: Cryptographic protocols – Decidability – Complexity – Transducers

1 Introduction

Cryptographic protocols are communication protocols that use cryptographic primitives, such as encryption and digital signatures, to achieve certain goals. For example, to buy and sell goods or to carry out bank transactions over the Internet, it is necessary that principals (e.g., the bank and the customer) authenticate each other, i.e., make sure that they are really talking to each other, and that a secret key is exchanged between the principals for secure communication. Well-known examples of such protocols are the Secure Socket Layer [24], Kerberos [28], and the Secure Shell [6].

It is very hard to prove cryptographic protocols secure or to find attacks on these protocols. This is because cryptographic protocols are supposed to be secure even in a "hostile" environment in which an intruder has control over the whole network. Such an intruder can insert new messages into the network, impersonate other principals, and read, modify, replay, and delete all messages sent over the network. In addition, several sessions of a protocol may run interleaved and the intruder can combine messages coming from different sessions.

For this reason informal arguments about the security of cryptographic protocols often turned out to be incomplete, and attacks on these protocols were discovered only years after they were proposed [29].

Formal methods. In contrast to informal arguments, formal methods are very successful in analyzing the security of cryptographic protocols. Using these methods, many flaws have been found in published protocols [14]. Most of these methods are based on the so-called Dolev-Yao model [18] – models such as special-purpose logics [12] or complexity theoretic and probabilistic models [7] are not discussed here.

In the Dolev-Yao model messages are represented as terms. For instance, the term $\mathsf{enc}_k(m)$ denotes the message m encrypted by k. The general assumption is that the cryptographic primitives used in a protocol work perfectly (*perfect cryptography assumption*): for instance, given the term $\mathsf{enc}_k(m)$ an intruder can get hold of m if and only if he knows k. Without k no partial information about m is leaked. Also, the intruder cannot guess keys or perform statistical tests.

By now, a large variety of different formalisms, methods, and tools for cryptographic protocol analysis based on the Dolev-Yao model is available [16, 32]. The formalisms include rewriting [13, 39], process algebras [1, 3, 19, 30, 40], first- and higher-order predicate logics [15, 37], and different special-purpose formalisms [16]. The methods and tools include those based on theorem proving [10, 15, 37], model checking [30, 34], and special-purpose algorithms for automatic analysis of protocols [3, 4, 9, 27, 41].

* Part of this work was carried out while the author was at Stanford University supported by the "Deutsche Forschungsgemeinschaft" (DFG).

Decidability. For different interesting classes of protocols and intruders, security has been shown to be decidable, where in this context security usually means secrecy. A protocol preserves secrecy if the intruder cannot get hold of a message that should only be known by the honest principals.

Decidability depends on the following parameters. The number of interleaved sessions considered in the analysis of a protocol may be bounded or unbounded; the size of messages exchanged between principals and produced by the intruder may be bounded or unbounded; and pairing of messages, nonces (random numbers modeled as fresh constants), and complex keys (complex terms used as keys) may or may not be allowed.

More specifically, if one allows for an unbounded number of sessions, security is in general undecidable [2, 3, 20, 21], except when the message size is bounded and nonces are disallowed (then security is EXPTIME-complete [20]), or pairing is disallowed (then security is in P [3, 17]). The situation is much better if one puts a bound on the number of sessions and disallows nonces. Then, even with pairing and unbounded message size, security is decidable [3, 27, 39] and NP-hard [3, 39]; Rusinowitch and Turuani [39] show NP-completeness with complex keys. As further discussed below, except for complex keys, we make similar assumptions in our models.

Open-ended data structures. Research on protocol analysis has concentrated mainly on protocols with *closed-ended data structures*, where messages exchanged between principals have fixed and finite format. In what follows, we will refer to these protocols as *closed-ended protocols*. However, in many protocols the data structures are *open-ended*: the number of data fields that must be processed by a principal in one receive-send action is unbounded, where *receive-send action* means that a principal receives a message and, after some internal computation, reacts by sending a message. We will call protocols with open-ended data structures *open-ended*.

One example of an open-ended protocol is the Internet Key Exchange protocol (IKE) [26], in which a principal needs to pick a *security association (SA)* among an a priori unbounded list of SAs, where an SA is the collection of algorithms and other information used for encryption and authentication. Such a list of SAs is an open-ended data structure since it has an unbounded number of data fields to be examined by a principal. An attack on IKE, found by Zhou [43] and, independently, Ferguson and Schneier [22], shows that when modeling open-ended protocols, the open-ended data structures must be taken into account, since otherwise some attacks might not be found. In other words, as also pointed out by Meadows [32], open-endedness is security relevant.

Other typical open-ended protocols are group protocols, for example, the recursive authentication protocol (RA protocol) [11] and A-GDH.2 [5], which is part of the CLIQUES project [42]. In the RA protocol a key distribution server receives, in one receive-send action, an a priori unbounded sequence of request messages, where each request message contains a pair of principals who want to share session keys. From this sequence the server needs to generate a corresponding sequence of certificates containing the session keys. These sequences are open-ended data structures: in one receive-send action the server needs to process an unbounded number of data fields, namely, the sequence of pairs of principals. In Appendix A, the RA protocol is described in detail.

To the best of our knowledge, the only contributions on formal analysis of open-ended protocols are the following. The RA protocol [11] has been analyzed by Paulson [36], using the Isabelle theorem prover, as well as by Bryans and Schneider [10], using the PVS theorem prover. The A-GDH.2 protocol [5] has been analyzed by Meadows [31] with NRL Analyzer and manually by Pereira and Quisquater [38] based on a model similar to the strand spaces model. Decidability issues for open-ended protocols have not been studied up until now.

Contribution of the paper. This paper addresses open-ended protocols, and thus deals with one of the challenges pointed out by Meadows [32]. The goal is to devise a protocol model rich enough to capture a large class of open-ended protocols such that security is decidable; the long-term goal is to develop tools for automatic verification of open-ended protocols.

As discussed above, open-ended protocols make it necessary to model principals who can perform in one receive-send action an unbounded number of *internal actions*; only then can they handle open-ended data structures. Thus, the first problem is to find a suitable computational model for receive-send actions.

It turns out that one cannot simply extend the existing models. More specifically, Rusinowitch and Turuani [39] describe receive-send actions by single rewrite rules and show security to be NP-complete. The rewrite rules may contain nonlinear terms (in particular, messages can be checked for equality and can be copied) and the principals have unbounded memory. To deal with open-ended protocols, we generalize their model in a straightforward way and show that if receive-send actions are described by sets of rewrite rules, security is undecidable. The reason for the undecidability is that principals can compare or copy messages of arbitrary size, which in particular is possible if they have unbounded memory. Therefore, we need a computational model in which principals have finite memory and cannot compare or copy messages of arbitrary size.

For this reason, we propose to use transducers, i.e., finite automata with output, as the computational model for receive-send actions, since transducers satisfy the above restrictions and can still deal with open-ended data structures. Section 4.1 contains a detailed discussion on our so-called transducer-based protocol model and Appendix B provides a description of the RA protocol in this model.

The main result of this paper is that in the transducer-based model, security is decidable and PSPACE-hard under the following assumptions: the number of sessions is bounded, i.e., a protocol is analyzed assuming a fixed number of interleaved protocol runs, and nonces and complex keys are not allowed. We, however, put no restrictions on the Dolev-Yao intruder; in particular, the message size is unbounded. These assumptions are quite close to those made in decidable models for closed-ended protocols [3, 8, 23, 27, 33, 39]; nevertheless, in some respects they are more restrictive. First, we note that in most models for closed-ended protocols, complex keys are allowed. Second, these models usually include asymmetric encryption, while here we only have symmetric encryption. However, extending our models and results to include asymmetric encryption seems straightforward. Finally, we remark that even though in models for closed-ended protocols nonces are in most cases not modeled explicitly, they can be simulated as the messages returned by principals have fixed and finite format, which is not the case for open-ended protocols.

The results presented here indicate that from a computational point of view, the analysis of open-ended protocols is harder than for closed-ended protocols, for which security is "only" NP-complete [39]. The additional complexity derives from the fact that now we have, besides the Dolev-Yao intruder, another source of infinite behavior: the unbounded number of internal actions, that is, paths in the transducers of unbounded length. This requires devising new proof techniques to show decidability. Roughly speaking, using that transducers only have finite memory we will use a pumping argument showing that the length of paths in the transducers can be bounded in the size of the problem instance.

Structure of the paper. In Sect. 2, we define a generic protocol model to describe open-ended protocols. In this model, receive-send actions can be arbitrary computations. In Sect. 3, we consider an instance of the generic model in which receive-send actions are specified by sets of rewrite rules and show the abovementioned undecidability result. Then, the transducer-based model, i.e., the instance of the generic protocol model in which receive-send actions are given by transducers, is introduced in Sect. 4. This section also contains the discussion mentioned above. Section 5 contains the actual decidability result, which is proved in the subsequent sections. Section 11 contains the proof of the complexity lower bound. Finally, we conclude in Sect. 12. Appendix A provides a detailed description of the RA protocol and in Appendix B a formalization of the key distribution server in the transducer-based protocol model is given.

2 A generic protocol model

The generic protocol model will serve in this paper as a general framework for protocol models. It has many decidable models for closed-ended protocols as instance. In particular, the underlying assumptions basically coincide with those in [3, 8, 27, 33, 39]. It also includes the models for open-ended protocols considered in the present work. The most important characteristic of the generic protocol model is that receive-send actions are, roughly speaking, binary relations over the message space and thus can be interpreted as *arbitrary* computations. The different instances of the generic protocol model can be distinguished in the kind of computational model used to describe receive-send actions. For example, receive-send actions could be described as rewrite rules [39] or transducers, as proposed in Sect. 4.

The main features of the generic protocol model can be summarized as follows:

- A generic protocol is described by a finite set of principals;
- The internal state space of a principal may be infinite (which, for example, enables a principal to store arbitrary long messages);
- Every principal is described by a finite sequence of receive-send actions;
- Receive-send actions are arbitrary computations.

We make the following assumptions:

- The intruder is the standard Dolev-Yao intruder; in particular, we do not put restrictions on the size of messages.
- Principals and the intruder cannot generate nonces.
- Keys are atomic messages, i.e., constants.
- The number of sessions is bounded. More precisely, the sessions considered in the analysis are only those encoded in the protocol description itself.

As mentioned earlier, except for the restriction to atomic keys, these are standard assumptions also made in decidable models for closed-ended protocols.

We now provide a formal definition of the generic protocol model.

2.1 Messages

Let $\mathcal{N}$ denote a finite set of *atomic messages* containing keys, names of principals, etc. as well as the special atomic message secret. The *set of messages* (over $\mathcal{N}$) is the least set $\mathcal{M}$ that satisfies the following properties:

- $\mathcal{N} \subseteq \mathcal{M}$;
- If $m, m' \in \mathcal{M}$, then $mm' \in \mathcal{M}$;
- If $m \in \mathcal{M}$ and $a \in \mathcal{N}$, then $\mathsf{enc}_a(m) \in \mathcal{M}$;
- If $m \in \mathcal{M}$, then $\mathsf{hash}(m) \in \mathcal{M}$.

Concatenation is an associative operation, and thus we require $(mm')m'' = m(m'm'')$. We point out that in many other approaches (see, e.g., [3, 8, 16, 20, 27, 39]) concatenation is modeled by a free binary operator, which in particular is not associative. Due to this simplification, in these models some attacks discovered in our model might not be detected. Conversely, other models are more accu-

rate in that they allow keys to be complex messages, while here we assume keys to be atomic messages.

Let ε denote the *empty message*. Then, define $\mathcal{M}_\varepsilon := \mathcal{M} \cup \{\varepsilon\}$. Note that ε is not allowed inside encryptions or hashes, that is, $\mathsf{enc}_a() \notin \mathcal{M}_\varepsilon$ and $\mathsf{hash}() \notin \mathcal{M}_\varepsilon$.

The *size* $|m|$ of a message m is the number of occurrences of atomic messages, encryption, and hash functions in m; $|\varepsilon| := 0$.

Later, we will also consider terms, i.e., messages with variables. Let V be a set of variables disjoint from $\mathcal{N}$. A *term* t over V (and $\mathcal{N}$) is a message over the atomic messages $\mathcal{N} \cup V$, where variables are not allowed as keys, i.e., terms of the form $\mathsf{enc}_v(\cdot)$ for some variable v are forbidden. We write $t(v_0, \ldots, v_{n-1})$ to say that the set of variables occurring in t is a subset of $\{v_0, \ldots, v_{n-1}\}$. Let $\mathcal{T}(V)$ denote the set of terms over V and $\mathcal{T}_\varepsilon(V) := \mathcal{T}(V) \cup \{\varepsilon\}$. We call a term t *linear* if every variable in t occurs at most once. Given $t(v_0, \ldots, v_{n-1})$ and terms $t_0, \ldots, t_{n-1}$, $t[v_0/t_0, \ldots, v_{n-1}/t_{n-1}]$ denotes the term obtained from t by simultaneously substituting the variables v_i by t_i. Sometimes we simply write $t[t_0, \ldots, t_{n-1}]$. A term $t' \in \mathcal{T}_\varepsilon(V)$ is a *subterm* of $t \in \mathcal{T}_\varepsilon(V)$ if there exists a linear term $t''(v)$ containing the variable v such that $t = t''[t']$. A *substitution* σ is a mapping from V into $\mathcal{M}_\varepsilon$. If $t \in \mathcal{T}_\varepsilon(V)$, then $\sigma(t)$ denotes the message obtained from t by replacing every variable v in t by $\sigma(v)$.

The *depth* $\mathsf{depth}(t)$ *of a term* t is the maximum number of nested hashes and encryptions in t, i.e.,

- $\mathsf{depth}(\varepsilon) := 0$, $\mathsf{depth}(a) := 0$ for every $a \in \mathcal{N} \cup V$;
- $\mathsf{depth}(tt') := \mathsf{max}\{\mathsf{depth}(t), \mathsf{depth}(t')\}$;
- $\mathsf{depth}(\mathsf{enc}_a(t)) := \mathsf{depth}(\mathsf{hash}(t)) := 1 + \mathsf{depth}(t)$.

Given a finite subset of messages $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$, the maximum depth of messages in $\mathcal{K}$ is denoted by $\mathsf{depth}(\mathcal{K}) := \mathsf{max}\{\mathsf{depth}(m) \mid m \in \mathcal{K}\}$.

2.2 The intruder model

We use the standard Dolev-Yao intruder [18]. That is, an intruder has complete control over the network and can derive new messages from his current knowledge by composing, decomposing, encrypting, decrypting, and hashing messages. We do not impose any restrictions on the size of messages.

The (possibly infinite) set $\mathsf{d}(\mathcal{K})$ of messages the intruder can derive from $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$ is the smallest set satisfying the following conditions:

- $\mathcal{K} \subseteq \mathsf{d}(\mathcal{K})$;
- If $mm' \in \mathsf{d}(\mathcal{K})$, then $m \in \mathsf{d}(\mathcal{K})$ and $m' \in \mathsf{d}(\mathcal{K})$ (decomposition);
- If $\mathsf{enc}_a(m) \in \mathsf{d}(\mathcal{K})$ and $a \in \mathsf{d}(\mathcal{K})$, then $m \in \mathsf{d}(\mathcal{K})$ (decryption);
- If $m \in \mathsf{d}(\mathcal{K})$ and $m' \in \mathsf{d}(\mathcal{K})$, then $mm' \in \mathsf{d}(\mathcal{K})$ (composition);
- If $m \in \mathsf{d}(\mathcal{K})$, $m \neq \varepsilon$, and $a \in \mathcal{N} \cap \mathsf{d}(\mathcal{K})$, then $\mathsf{enc}_a(m) \in \mathsf{d}(\mathcal{K})$ (encryption);
- If $m \in \mathsf{d}(\mathcal{K})$ and $m \neq \varepsilon$, then $\mathsf{hash}(m) \in \mathsf{d}(\mathcal{K})$ (hashing).

Let $\mathsf{an}(\mathcal{K})$ denote the closure of $\mathcal{K}$ under decomposition and decryption, and $\mathsf{syn}(\mathcal{K})$ the closure of $\mathcal{K}$ under composition, encryption, and hashing. It easy to see that, since we only allow for atomic keys, $\mathsf{d}(\mathcal{K})$ can be obtained by first taking the closure of $\mathcal{K}$ under decomposition and decryption, and from this the closure under composition, encryption, and hashing. Formally (see, for instance, [37] for a proof):

Lemma 1. $\mathsf{d}(\mathcal{K}) = \mathsf{syn}(\mathsf{an}(\mathcal{K}))$ *for every* $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$.

We point out that in case of complex keys, i.e., if terms of the form $\mathsf{enc}_m(m')$ are allowed where both m and m' are arbitrary messages, the lemma does not hold since, for instance, to derive m' from $\mathsf{enc}_m(m')$ it may be necessary to construct m before decrypting $\mathsf{enc}_m(m')$ using m [33]. The decidability result presented in this paper will make use of the above lemma.

2.3 Protocols

Protocols are described by sets of principals, and every principal is defined by a sequence of receive-send actions, which are performed one after the other. Since we are interested in attacks, the definition of a protocol also contains the initial intruder knowledge. Formally, principals and protocols are defined as follows.

Definition 1. *A* generic principal Π *is a tuple* (Q, I, n, α)*, where*

- Q *is the (possibly infinite) set of states of* Π*;*
- I *is the set of initial states of* Π*;*
- n *is the number of receive-send actions to be performed by* Π*;*
- α *is a mapping assigning to every* $j \in \{0, \ldots, n-1\}$ *a receive-send action* $\alpha(j) \subseteq Q \times \mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon \times Q$.

A generic protocol P *is a tuple* $(\{\Pi_i\}_{i<l}, \mathcal{K})$*, where*

- $\{\Pi_i\}_{i<l}$ *is a family of* l *generic principals, and*
- $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$ *is the initial intruder knowledge.*

Note that receive-send actions are arbitrary relations. Intuitively, they take an input message (2. component) and nondeterministically, depending on the current state (1. component), return an output message (3. component) plus a new state (4. component). Later, when we consider instances of the generic protocol model, one receive-send action of a principal will consist of an unbounded number of internal actions. By allowing receive-send actions to be nondeterministic and principals to have a set of initial states, instead of a single initial state, one can model more flexible principals: for instance, those that nondeterministically choose one principal who they want to talk to or one SA from the list of SAs in IKE.

2.4 Attacks on protocols

In an attack on a protocol, the receive-send actions of the principals are interleaved in some way and the intruder,

who has complete control over the communication, tries to produce inputs for the principals such that from the corresponding outputs and his initial knowledge he can derive the secret message secret. Formally, an attack is defined as follows.

Definition 2. *Let $P = (\{\Pi_i\}_{i<l}, \mathcal{K})$ be a generic protocol with $\Pi_i = (Q_i, I_i, n_i, \alpha_i)$, for $i < l$. An* attack *on P is a tuple consisting of the following components:*

- *A total ordering $<$ on the set $\{(i,j) \mid i < l, j < n_i\}$ such that $(i,j) < (i,j')$ implies $j < j'$ (the execution order of the receive-send actions);*[1]
- *A mapping ψ assigning to every (i,j), $i < l$, $j < n_i$, a tuple*

$$\psi(i,j) = (q_i^j, m_i^j, m'^j_i, q_i^{j+1})$$

with
  - *$q_i^j, q_i^{j+1} \in Q_i$ (the state of Π_i before and after performing $\alpha_i(j)$); and*
  - *$m_i^j, m'^j_i \in \mathcal{M}_\varepsilon$ (the input message received and output message sent by the action $\alpha_i(j)$);*

such that

- *$q_i^0 \in I_i$ for every $i < l$;*
- *$m_i^j \in \mathsf{d}(\mathcal{K} \cup \{m'^{j'}_{i'} \mid (i',j') < (i,j)\})$ for every $i < l$, $j < n_i$;*
- *$(q_i^j, m_i^j, m'^j_i, q_i^{j+1}) \in \alpha_i(j)$ for every $i < l$, $j < n_i$.*

An attack is called successful *if* $\mathsf{secret} \in \mathsf{d}(\mathcal{K} \cup \{m'^j_i \mid i < l, j < n_i\})$.

The decision problem we are interested in is the following:

ATTACK: Given a protocol P, decide whether there exists a successful attack on P.

A protocol guarantees *secrecy* if there does not exist a successful attack. In this case, we say that the protocol is *secure*.

Whether ATTACK is decidable or not heavily depends on what kinds of receive-send actions a principal is allowed to perform. In the following sections, we look at different instances of generic protocols, i.e., different computational models for receive-send actions, and study the problem ATTACK for the classes of protocols thus obtained.

3 The undecidability result

We extend the model proposed by Rusinowitch and Turuani [39] in a straightforward way such that open-ended protocols can be handled, and we show that this extension leads to undecidability of security.

[1] Although we assume a linear ordering on the receive-send actions performed by a principal, we could just as well allow partial orderings (as in [39]) without any impact on the decidability and complexity results.

The model by Rusinowitch and Turuani can be considered as the instance of the generic protocol model in which receive-send actions are described by single rewrite rules of the form $t \to t'$, where t and t' are terms.[2] The internal state of a principal is given implicitly by the values assigned to the variables occurring in the rewrite rules – different rules may share variables. In particular, a principal has unbounded memory to store information for use in subsequent receive-send actions. Roughly speaking, a message m is transformed by a receive-send action of the form $t \to t'$ into the message $\sigma(t')$, where σ is a substitution satisfying $m = \sigma(t)$. In [39], it is shown that in this setting, ATTACK is NP-complete.

Of course, in this model open-ended data structures cannot be handled since the left-hand side t of a rewrite rule has a fixed and finite format, and thus one can only process messages with a fixed number of data fields.

A straightforward and rather naïve extension of this model, which allows one to deal with open-ended data structures, is to describe receive-send actions by sets of rewrite rules, which can nondeterministically be applied to the input message. This extension yields what we call the rule-based protocols.

More specifically, a receive-send action in a rule-based protocol will be defined by a set of input, output, and so-called process rules. A message is processed by such an action as follows. First, one of the input rules is applied, resulting in a new message. Then, Nondeterministically, process rules are applied to this message, and, finally, one of the output rules is used to produce the actual output. In a more powerful model, one could allow a principal to produce output when applying a process rule as well. However, to obtain a stronger undecidability result, we stick to the more restricted model.

Formally, as already mentioned, a *rewrite rule* is of the form $t \to t'$, where t and t' are terms. A *(rule-based) action* A of a principal is a tuple (I, O, R), where I, O, and R are finite sets of rewrite rules (the input, output, and process rules, respectively). For every rule $t \to t' \in R$ we require that for all substitutions σ, $|\sigma(t')| < |\sigma(t)|$. This guarantees that process rules can only be applied to a message a finite number of times.

A rule-based action A defines the following binary relation R_A on $\mathcal{M}_\varepsilon$: $(m, m') \in R_A$ iff there exist substitutions $\sigma_0, \ldots, \sigma_n$ and rewrite rules $r_0, \ldots, r_n$ with $r_i = t_i \to t'_i$, for every $i \le n$, such that

- $r_0 \in I$, $r_n \in O$, and $r_i \in R$ for every $0 < i < n$;
- $\sigma_0(t_0) = m$; $\sigma_n(t'_n) = m'$; and
- $\sigma_i(t'_i) = \sigma_{i+1}(t_{i+1})$ for every $i < n$.

A generic protocol for which the receive-send actions can be described by rule-based actions is called a *rule-based protocol*. Note that, since in this setting principals do not

[2] Since Rusinowitch and Turuani allow for complex keys, the terms are more general than the ones we use here. However, we will only consider terms as defined in Sect. 2.1.

have states, receive-send actions are simply subsets of $\mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon$ instead of $Q \times \mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon \times Q$.

Theorem 1. ATTACK *is undecidable for rule-based protocols.*

The proof is rather straightforward. It is by reduction from *Post's correspondence problem* (PCP), which is defined as follows. Given an alphabet Σ with at least two letters and two sequences $u_0, \dots, u_{n-1}$ and $v_0, \dots, v_{n-1}$ of words over Σ (including the empty word ε), decide whether there exist indices $i_0, \dots, i_{k-1}$, $k > 0$, with $u_{i_0} \dots u_{i_{k-1}} = v_{i_0} \dots v_{i_{k-1}}$.

Given an instance $\mathcal{I}$ of PCP, we define the corresponding rule-based protocol P as follows. We construct P in such a way that the intruder only obtains a secret if he can guess a solution of $\mathcal{I}$. Formally, P has one principal performing one action $A = (I, O, R)$ with

- $I := \{x \to x = x\}$;
- $O := \{u_i = v_i \to \mathsf{secret} \mid i < k\}$;
- $R := \{u_i x = v_i y \to x = y \mid i < k\}$,

where x and y are variables. The initial intruder knowledge is $\mathcal{K} := \Sigma \cup \{=\}$. Now it is easy to see that P allows a successful attack iff the instance of PCP has a solution.

In this reduction we merely used that a principal can copy messages (see the input rule). A reduction to PCP is also possible if nonlinear terms may only occur on the left-hand side of a rewrite rule, and thus principals can only test submessages for equality. We also remark that, even when we restrict rewrite rules to contain only linear terms but allow a principal to store one message and compare this message to the message currently processed, this again allows equality tests and, therefore, leads to undecidability.

4 Transducer-based protocol model

As just discussed, when, roughly speaking, principals can process open-ended data structures and, in addition, can compare submessages of arbitrary size for equality, copy messages, or having unbounded memory, then security is undecidable. To obtain decidability, we need a device with only finite memory and that does not allow one to compare or copy messages of arbitrary size. One such device is a transducer, i.e., a finite automaton with output.

In what follows, we define the instance of the generic protocol model in which receive-send actions are described by transducers. Section 4.1 contains a discussion of this model.

If Σ is a finite alphabet, Σ^* will denote the set of finite words over Σ, including the empty word ε; $\Sigma^+ := \Sigma^* \setminus \{\varepsilon\}$.

Definition 3. *A* transducer $\mathcal{A}$ *is a tuple* $(Q, \Sigma, \Omega, I, \Delta, F)$ *where*

- Q *is the finite set of states of* $\mathcal{A}$*;*
- Σ *is the finite input alphabet;*
- Ω *is the finite output alphabet;*
- $I \subseteq Q$ *is the set of initial states of* $\mathcal{A}$*;*
- $\Delta \subseteq Q \times \Sigma^* \times \Omega^* \times Q$ *is the finite set of transitions of* $\mathcal{A}$*; and*
- $F \subseteq Q$ *is the set of final states of* $\mathcal{A}$*.*

A *path* π (of length n) in $\mathcal{A}$ from p to q is of the form $q_0(v_0, w_0)q_1 \dots (v_{n-1}, w_{n-1})q_n$ with $q_0 = p$, $q_n = q$, and $(q_i, v_i, w_i, q_{i+1}) \in \Delta$ for every $i < n$; π is called *strict* if $n > 0$, and v_0 and v_{n-1} are nonempty words. The word $v_0 \dots v_{n-1}$ is the *input label* and $w_0 \dots w_{n-1}$ is the *output label* of π. A path of length 0 has input and output label ε. We write $p(v, w)q \in \mathcal{A}$ ($p(v, w)q \in_s \mathcal{A}$) if there exists a (strict) path from p to q in $\mathcal{A}$ with input label v and output label w.

If $S, T \subseteq Q$, then $\mathcal{A}(S, T) := \{(p, v, w, q) \mid p \in S, q \in T, p(v, w)q \in \mathcal{A}\} \subseteq Q \times \Sigma^* \times \Omega^* \times Q$. The *output of* $\mathcal{A}$ *on input* $v \in \Sigma^*$ is defined by $\mathcal{A}(v) := \{w \mid$ there exist $p \in I$ and $q \in F$ with $(p, v, w, q) \in \mathcal{A}(I, F)\}$.

If $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Omega \cup \{\varepsilon\}) \times Q$, then $\mathcal{A}$ is called *transducer with letter transitions* in contrast to transducers with word transitions. The following lemma shows that it suffices to consider transducers with letter transitions. The proof is straightforward.

Lemma 2. *Let* $\mathcal{A} = (Q, \Sigma, \Omega, I, \Delta, F)$ *be a transducer. Then there exists a transducer* $\mathcal{A}' = (Q', \Sigma, \Omega, I, \Delta', F)$ *with letter transitions such that* $Q \subseteq Q'$ *and for every* $S, T \subseteq Q$*:* $\mathcal{A}'(S, T) = \mathcal{A}(S, T)$.

In order to specify the receive-send actions of a principal, we consider special transducers, so-called message transducers, that satisfy certain properties. Message transducers interpret messages as words over the finite alphabet $\Sigma_\mathcal{N}$, consisting of the atomic messages as well as the letters "$\mathsf{enc}_a($", "$\mathsf{hash}($", and "$)$", that is,

$$\Sigma_\mathcal{N} := \mathcal{N} \cup \{\mathsf{enc}_a(\mid a \in \mathcal{N}\} \cup \{\mathsf{hash}(,)\}.$$

Messages considered as words over $\Sigma_\mathcal{N}$ have always a balanced number of opening parentheses, i.e., "$\mathsf{enc}_a($" and "$\mathsf{hash}($", and closing parentheses, i.e., "$)$". Often these letters will occur in expressions in the text (as in the definition of $\Sigma_\mathcal{N}$) without matching opening and closing parentheses.

A message transducer reads a message (interpreted as a word) from left to right, thereby producing some output. If messages are considered as finite trees (where leaves are labeled with atomic messages and internal nodes are labeled with the encryption or hash symbol), a message transducer traverses such a tree from top to bottom and from left to right.

We assume that if a message transducer reads a message, then the resulting outputs are messages rather than arbitrary words. A message transducer has to meet this condition both "externally" (see condition 1) and "internally" (condition 2).

Definition 4. *A* message transducer $\mathcal{A}$ *(over* $\mathcal{N}$*) is a tuple* $(Q, \Sigma_{\mathcal{N}}, I, \Delta, F)$ *such that* $(Q, \Sigma_{\mathcal{N}}, \Sigma_{\mathcal{N}}, I, \Delta, F)$ *is a transducer with letter transitions, and*

1. *For every* $x \in \mathcal{M}_\varepsilon$, $\mathcal{A}(x) \subseteq \mathcal{M}_\varepsilon$*; and*
2. *For all* $p, q \in Q$, $x \in \mathcal{M}$, *and* $y \in \Sigma_{\mathcal{N}}^*$, *if* $p(x, y)q \in_s \mathcal{A}$, *then* $y \in \mathcal{M}_\varepsilon$.

It remains to investigate whether conditions 1 and 2 are decidable. Nevertheless, for the transducer describing the server in the recursive authentication protocol (Appendix B), it is easy to see that these conditions are satisfied.

For $S, T \subseteq Q$, we define $M_{\mathcal{A}}(S, T) := \mathcal{A}(S, T) \cap (Q \times \mathcal{M}_\varepsilon \times \Sigma_{\mathcal{N}}^* \times Q)$. By the definition of message transducers, $M_{\mathcal{A}}(I, F) \subseteq (Q \times \mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon \times Q)$ if I is the set of initial states and F is the set of final states of $\mathcal{A}$. Thus, message transducers specify receive-send actions of principals (in the sense of Definition 1) in a natural way.

In order to define one principal (i.e., the whole sequence of receive-send actions a principal performs) by a single transducer, we consider extended message transducers: $\mathcal{A} = (Q, \Sigma_{\mathcal{N}}, \Delta, (I_0, \ldots, I_n))$ is an *extended message transducer* if $\mathcal{A}_{I_j, I_{j+1}} := (Q, \Sigma_{\mathcal{N}}, I_j, \Delta, I_{j+1})$ is a message transducer for all $j < n$. Given such an extended message transducer, it defines the principal (Q, I_0, n, α), where $\alpha(j) = M_{\mathcal{A}_{I_j, I_{j+1}}}(I_j, I_{j+1})$ for $j < n$. In this setting, an internal action of a principal corresponds to applying one transition in the extended message transducer.

Definition 5. *A* transducer-based protocol P *is a generic protocol where the principals are defined by extended message transducers.*

4.1 Discussion of the transducer-based protocol model

We discuss capabilities and limitations of the transducer-based protocol model. To this end, we compare this model with the models usually used for closed-ended protocols. To make the discussion more concrete, we concentrate on the model proposed by Rusinowitch and Turuani (see Sect. 3), subsequently called *rewriting model*, a model that contains the main features – we also could have used models proposed, for instance, in [3, 8, 27, 33]. In the introduction, differences regarding basic assumptions such as complex keys and nonces have been discussed already. Here we concentrate on the main differences between the two models with respect to the way receive-send actions are described. In the rewriting model, receive-send actions are single rewrite rules, while in the transducer-based model they are message transducers.

Let us explain the capabilities of message transducers compared to rewrite rules.

Open-ended data structures. As mentioned in Sect. 3, with a single rewrite rule one cannot process an unbounded number of data fields. This is, however, possible with transducers.

For example, considering IKE (see introduction), it is easy to specify a transducer that (i) reads a list of SAs, each given as a sequence of atomic messages, (ii) picks one SA, and (iii) returns it. With a single rewrite rule, one could not parse the whole list of SAs. One can, however, consider this problem as matching modulo associativity. Unfortunately, as mentioned in Sect. 2.1, in models for closed-ended protocols, concatenation is usually considered a free binary operator, and thus this kind of matching is not supported.

The transducer-based model of the RA protocol (Appendix B) shows that transducers can also handle more involved open-ended data structures: the key distribution server in this protocol generates a sequence of certificates from a *request message* of the form

$$\mathsf{hash}(m_0 \mathsf{hash}(m_1 \cdots \mathsf{hash}(m_n) \cdots)\,,$$

where the m_i's are sequences of atomic messages and the nesting depth of the hashes is a priori unbounded (see Appendices A and B for the exact definition of the messages).

Clearly, a transducer cannot match opening and closing parenthesis if they are nested arbitrarily deep since messages are interpreted as words. However, often this is not necessary: in IKE, the list of SAs is a message without any nesting. In the RA protocol, the structure of request messages is very simple and can be parsed by a transducer. Note that a transducer does not need to check whether the number of closing parentheses in the request message matches the number of hashes because all words sent to a message transducer are sent by the intruder, who can only deal with messages, and thus well-formed words.

Simulating rewrite rules. Transducers can simulate certain receive-send actions described by single rewrite rules. Consider, for example, the rule $\mathsf{enc}_k(x) \to \mathsf{hash}(kx)$, where x is a variable and k an atomic message: first, the transducer would read "$\mathsf{enc}_k($" and output "$\mathsf{hash}(k$", and then reads, letter by letter, the rest of the input message, i.e., "$x)$" – more precisely, the message substituted for x – and simultaneously writes it into the output. The transducer can also check whether the last letter of the input is a closing parenthesis; again, this is not necessary because all words sent to a message transducer (by the intruder) are well formed.

We now turn to the limitations of the transducer-based model compared to the rewriting model. The main limitations are the following:

1. *Finite memory:* In the rewriting model, principals can store messages of arbitrary size to use them in subsequent receive-send actions. This is not possible with transducers since they only have finite memory.
2. *Comparing messages:* In the rewriting model, principals can check whether submessages of the input mes-

sage coincide; for example, if $t = \mathsf{hash}(kx)x$, with k an atomic message and x a variable, a principal can check whether plain text and hash match. Transducers cannot do this.

3. *Copying messages:* In the rewriting model, principals can copy messages of arbitrary size; for example, in the rule $\mathsf{enc}_k(x) \to \mathsf{hash}(kx)x$, the message x is copied. Again, a transducer would need to store x in some way, which is not possible because of the finite memory. As illustrated above, a transducer could, however, simulate a rule such as $\mathsf{enc}_k(x) \to \mathsf{hash}(kx)$.
4. *Linear terms:* A transducer cannot simulate all rewrite rules with linear left- and right-hand side. Consider, for example, the rule $\mathsf{enc}_k(xAy) \to \mathsf{hash}(yAx)$, where x and y are variables and A is an atomic message. Since in the output the order of x and y is switched, a transducer would have to store the messages substituted for x and y. However, this requires unbounded memory.

The undecidability result presented in Sect. 3 indicates that, if open-ended data structures are involved, restrictions 1, 2, and 3 seem to be unavoidable. However, in Sect. 3 we extended the rewriting model in a quite naïve way. One question is whether there are better suited extensions. Also, it remains to be seen whether the last restriction can be lifted. Future work will investigate to what extent *tree* transducers can help here and if so what kinds of tree transducers are best suited. In addition, it might be possible to combine different computational models for receive-send actions. For instance, a hybrid model in which some receive actions are described by rewrite rules and others by transducers might still be decidable.

The model of the RA protocol (presented in Appendix B) illustrates that, even with the restrictions listed above, some protocols, including those not amenable to models for closed-ended protocols, can be captured within the transducer-based protocol model. Typing of messages, which for example means that a principal accepts a message only if principal names and nonces are atomic messages, often helps to overcome the restrictions. For instance, assuming typed messages in the RA protocol allows one to model a server with finite memory.

5 Decidability result

We prove the following theorem.

Theorem 2. ATTACK *is decidable for transducer-based protocols.*

Obviously, it suffices to show that the following problem is decidable since in attacks interleavings of receive-send actions and initial and final states of the message transducers can be guessed.

PATHPROBLEM. *Given a finite set* $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$ *and message transducers* $\mathcal{A}_0, \dots, \mathcal{A}_{k-1}$, $k \geq 0$, *with*

$$\mathcal{A}_i = (Q_i, \Sigma_\mathcal{N}, \{q_i^I\}, \Delta_i, \{q_i^F\}),$$

decide whether there exist messages $m_i, m_i' \in \mathcal{M}_\varepsilon$, $i < k$ *such that*

1. $m_i \in \mathsf{d}(\mathcal{K} \cup \{m_0', \dots, m_{i-1}'\})$ *for every* $i < k$,
2. $q_i^I(m_i, m_i')q_i^F \in \mathcal{A}_i$ *for every* $i < k$, *and*
3. $\mathsf{secret} \in \mathsf{d}(\mathcal{K} \cup \{m_0', \dots, m_{k-1}'\})$.

We write an instance of the PATHPROBLEM as

$$(\mathcal{K}, \mathcal{A}_0, \dots, \mathcal{A}_{k-1})$$

and a solution of such an instance as a tuple

$$(m_0, m_0', \dots, m_{k-1}, m_{k-1}')$$

of messages. The size of instances is defined as the size of the representation for $\mathcal{K}$ and $\mathcal{A}_0, \dots, A_{k-1}$.

To show decidability, we have to cope with two sources of infinite behavior. First, the intruder can perform an unbounded number of steps to derive a new message (m_i or secret). Second, to perform one receive-send action, a principal can carry out an unbounded number of internal actions, i.e., the length of the paths $q_i^I(m_i, m_i')q_i^F \in \mathcal{A}_i$ is unbounded. Note that, because transducers may have ε-transitions, i.e., not in every transition is a letter read from the input, the number of transitions taken in one receive-send action is not even bounded in the size of the input message.

While the former source of infinite behavior was already present in the (decidable) models for closed-ended protocols [3, 27, 39], the latter source is new. To prove Theorem 2, one therefore needs to show not only that the number of actions performed by the intruder can be bounded, but also that the number of internal actions of principals can be bounded. In fact, it suffices to show the latter: if we can bound the number of internal actions, a principal only reads messages of bounded length and therefore the intruder need only produce messages of size bounded by this length.

To bound the number of internal actions, we apply a pumping argument showing that long paths in a message transducer can be truncated. This argument uses the fact that principals (the extended message transducers describing them) have only finite memory. More precisely, we show that in order to find the messages m_i, m_i' for every $i < k$, it suffices to consider paths from q_i^I to q_i^F in $\mathcal{A}_i$ bounded in length by the size of the problem instance; the argument will also show that the bounds can be computed effectively. Thus, a decision procedure can enumerate all paths of length restricted by the (computed) bound and check whether their labels satisfy the conditions. [Note that for every message m and finite set $\mathcal{K}' \subseteq \mathcal{M}_\varepsilon$, $m \in \mathsf{d}(\mathcal{K}')$ can be decided.] In particular, as a "by-product" our decision procedure will yield an actual attack (if any).

The pumping argument: To show that paths can be truncated, we first define a *solvability preserving (quasi-)ordering* on messages, which allows us to replace single messages in the intruder knowledge by new ones such that if in the original problem a successful attack exists, then it exists in the modified problem as well. This reduces the pumping argument to the following problem: Truncate paths in message transducers in such a way that the output of the original path is equivalent (w.r.t. the solvability preserving ordering) to the output of the truncated path. Thus, it remains to find criteria for truncating paths in this way. To this end, we introduce another quasiordering, the so-called *path truncation ordering*, which indicates at which positions a path can be truncated. To actually obtain a bound on the length of paths, one then needs to show that the equivalence relation induced by the path truncation ordering has a finite index – more accurately, an index that can be bounded in the size of the problem instance.

In what follows, the argument is described in more detail. The formal definition of the orderings are, however, postponed to later sections in order to focus on the main ideas.

Preserving solvability of instances of the path problem (cf. Sect. 6). For every $i \leq k$, we define a quasiordering[3] on messages $\preceq_i$ (the so-called solvability preserving ordering) that depends on the transducers $A_i, \ldots, A_{k-1}$ and has the following property (Proposition 1): For every solvable instance $(\mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$ of the path problem, every $m \in \mathcal{K}$ and $\overline{m} \in \mathcal{M}_\varepsilon$ with $m \preceq_i \overline{m}$, the instance $((\mathcal{K} \setminus \{m\}) \cup \{\overline{m}\}, \mathcal{A}_i, \ldots, A_{k-1})$ is solvable as well.

Having established this property, assume that a path $q_i^I(m_i, m_i')q_i^F \in \mathcal{A}_i$ is replaced by a shorter path such that the corresponding input and output labels of the shorter path, say, $\overline{m}_i$ and $\overline{m}_i'$, satisfy

$$\overline{m}_i \in \mathsf{d}(\mathcal{K} \cup \{m_0', \ldots, m_{i-1}'\})$$

and $m_i' \preceq_{i+1} \overline{m}_i'$. Then, after $\mathcal{A}_i$ has returned $\overline{m}_i'$ on input $\overline{m}_i$, the resulting intruder knowledge is $\overline{\mathcal{K}} := \mathcal{K} \cup \{m_0', \ldots, m_{i-1}', \overline{m}_i'\}$, i.e., m_i' was replaced by $\overline{m}_i'$. Using Proposition 1, we conclude that there still exists a solution for the rest of the instance, i.e., $(\overline{\mathcal{K}}, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_{k-1})$.

Consequently, it remains to find criteria for truncating long paths $q_i^I(m_i, m_i')q_i^F \in \mathcal{A}_i$ to shorter ones $q_i^I(\overline{m}_i, \overline{m}_i')q_i^F \in \mathcal{A}_i$ such that

1. $\overline{m}_i \in \mathsf{d}(\mathcal{K} \cup \{m_0', \ldots, m_{i-1}'\})$ and
2. $m_i' \preceq_{i+1} \overline{m}_i'$.

Truncating paths such that the first criterion is satisfied is rather easy. The involved part is to guarantee the second criterion. To this end, we introduce the path truncation ordering that indicates at which positions a path can be truncated such that criterion 2 is satisfied.

[3] A reflexive and transitive ordering

Truncating paths (cf. Sect. 9). We extend $\preceq_i$ to a quasiordering $\preceq_i^l$ (the path truncation ordering) on so-called left half-messages. Left half-messages are prefixes of messages (considered as words over $\Sigma_\mathcal{N}$). In particular, left half-messages may lack some closing parentheses. The "l" in $\preceq_i^l$ is the number of missing parentheses (the *level* of left half-messages); $\preceq_i^l$ only relates left half-messages of level l. Analogously, right half-messages are suffixes of messages. Thus, they may have too many closing parentheses; the number of additional parentheses determines the level of right half-messages. The equivalence relation $\equiv_i^l$ on left half-messages corresponding to $\preceq_i^l$ has the following property (Proposition 5): For all left half-messages α, α' of level l and right half-messages γ of level l, $\alpha \equiv_i^l \alpha'$ implies $\alpha\gamma \equiv_i \alpha'\gamma$. (Note that $\alpha\gamma$ and $\alpha'\gamma$ are messages.)

Now consider two positions $x < y$ in the path $\pi = q_i^I(m_i, m_i')q_i^F) \in \mathcal{A}_i$ such that α_x, α_y are the output labels up to these positions and γ_x, γ_y are the output labels beginning at these positions, i.e., $m_i' = \alpha_x\gamma_x = \alpha_y\gamma_y$. Clearly, α_x, α_y are left half-messages and γ_x, γ_y are right half-messages. Assume that α_x, α_y have the same level l (in particular, γ_x, γ_y have level l) and $\alpha_x \equiv_{i+1}^l \alpha_y$. Then, Proposition 5 implies that $m_i' = \alpha_y\gamma_y \equiv_{i+1} \alpha_x\gamma_y =: \overline{m}_i'$, where $\overline{m}_i'$ is the output label of the path obtained by cutting out the subpath in π between x and y.[4] Thus, $\equiv_{i+1}^l$ provides us with the desired criterion for truncating paths such that condition 2 is satisfied. In order to conclude that the length of paths can be bounded in the size of the problem instance, it remains to show that l and the index of $\equiv_{i+1}^l$ (i.e., the number of equivalence classes modulo $\equiv_{i+1}^l$ on left half-messages of level l) can be bounded in the size of the problem instance. To this end, the following point is demonstrated.

Bounding the depth of messages (cf. Sects. 7 and 8). We show that for every solvable instance

$$\mathcal{I} = (\mathcal{K}, \mathcal{A}_0, \ldots, \mathcal{A}_{k-1})$$

of the PathProblem there exists a solution

$$(m_0, m_0', \ldots, m_{k-1}, m_{k-1}')$$

such that the depth of the messages m_i, m_i', $i < k$, can be bounded in the size of $\mathcal{I}$ (Corollary 3). To this end, we first show that the depth of input messages can be bounded (Sect. 7) and then that the depth of output messages can be bounded in the depth of input messages (Sect. 8).

It follows that l is bounded, and one can also show that the index of $\equiv_{i+1}^l$ is bounded (Proposition 6). Therefore, $\equiv_{i+1}^l$ can serve as a finite criterion for truncating paths.

Following these steps, we now provide a detailed proof. We will define the different orderings and prove the properties needed. In Sect. 10 everything will be put together to show decidability of the path problem.

[4] One minor technical problem is that $\alpha_x\gamma_y$ does not need to be a message since it may contain a word of the form $\mathsf{enc}_a()$, which is not a message. However, if one considers three positions $x < y < z$, then one can show that either $\alpha_x\gamma_y$ or $\alpha_y\gamma_z$ is a message.

In order to simplify the presentation, we will not consider hashing, i.e., from now on, $\Sigma_{\mathcal{N}}$ does not contain the symbol "$\mathsf{hash}($". All definitions and results, however, easily carry over to this more general case.

6 Solvability preserving ordering

In this section, we formally define the solvability preserving ordering $\preceq_i$. In Sect. 6.2 we show that $\preceq_i$ is in fact solvability preserving. To do so, we first need to prove that $\preceq_i$ is closed under substitution (Sect. 6.1). Finally, it is shown that the index of the equivalence relation induced by $\preceq_i$ is finite (Sect. 6.3).

For messages $m_0, \dots, m_{n-1}$, and $\mathcal{K}, \mathcal{K}' \subseteq \mathcal{M}$ we write $\mathsf{an}(m_0, \dots, m_{n-1}, \mathcal{K})$ instead of $\mathsf{an}(\{m_0, \dots, m_{n-1}\} \cup \mathcal{K})$ and $\mathsf{an}_{\mathcal{K}'}(m_0, \dots, m_{n-1}, \mathcal{K})$ instead of $\mathsf{an}(\{m_0, \dots, m_{n-1}\} \cup \mathcal{K}) \cap \mathcal{K}'$. To define $\preceq_i$, we need the ordering $\preceq$.

Definition 6. *For messages $m, m' \in \mathcal{M}_\varepsilon$, define: $m \preceq m'$ iff $\mathsf{an}_{\mathcal{N}}(m, N) \subseteq \mathsf{an}_{\mathcal{N}}(m', N)$ for all $N \subseteq \mathcal{N}$.*

Intuitively, $m \preceq m'$ says that in every context the set of atomic messages derivable from m is a subset of the atomic messages derivable from m'. For example, for $a, a', a'' \in \mathcal{N}$, $\mathsf{enc}_a(\mathsf{enc}_{a'}(a'')) \preceq a'\mathsf{enc}_{a'}(\mathsf{enc}_a(a''))$.

Obviously, $\preceq$ is a quasiordering, i.e., it is reflexive and transitive.

The relation $\preceq_i$, $i \leq k$ depends on the message transducers $\mathcal{A}_i, \dots, \mathcal{A}_{k-1}$ and uses the relation $\sqsubseteq_i$. To understand the definition of $\preceq_i$, recall that we want to guarantee that if $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$ has a solution, say, $(m_i, m_i', \dots, m_{k-1}, m_{k-1}')$, then so does $(\mathcal{K} \setminus \{m\} \cup \{m'\}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$. The message m_i may have been constructed from submessages x of m. Now, if m is replaced by m', we need to make sure that in m' there is a submessage x' that can be used instead of x. This is why we need condition 2 in the definition below. We also need that $\preceq_i$ refines $\preceq_{i+1}$ (condition 1). Finally, in order to show that $\preceq_i$ is closed under substitution, condition 3 is required. In what follows, the relations $\preceq_i$ and $\sqsubseteq_i$ are defined recursively and mutually.

Definition 7. *For every $i \leq k$, the* solvability preserving ordering *$\preceq_i$ is defined as follows. For all $m, m' \in \mathcal{M}_\varepsilon$: $m \preceq_i m'$ iff (i) $m = m' = \varepsilon$ or (ii) $m \neq \varepsilon$, $m' \neq \varepsilon$, $m \preceq m'$, and if $i < k$, then*

1. *$m \preceq_{i+1} m'$,*
2. *For every $N \subseteq \mathcal{N}$ and $x \in \mathsf{an}(m, N)$ with $x = \mathsf{enc}_a(z)$ for some $a \in \mathcal{N}$ and $z \in \mathcal{M}$, there exists $x' := \mathsf{enc}_{a'}(z') \in \mathsf{an}(m', N)$ for some $a' \in \mathcal{N}$ and $z' \in \mathcal{M}$ such that $x \sqsubseteq_i x'$, and*
3. *$m \sqsubseteq_i m'$.*

For $i < k$ and messages $m, m' \in \mathcal{M}_\varepsilon$, we define $m \sqsubseteq_i m'$ iff (i) $m = m' = \varepsilon$ or (ii) $m \neq \varepsilon$, $m' \neq \varepsilon$, and for every $p, q \in Q_i$ and $y \in \mathcal{M}_\varepsilon$, $p(m, y)q \in_s \mathcal{A}_i$ implies that there exists $y' \in \mathcal{M}_\varepsilon$ with $p(m', y')q \in_s \mathcal{A}_i$ and $y \preceq_{i+1} y'$.

One easily shows by induction on $i \leq k$:

Lemma 3. *For every $i < k$, $\preceq_k$, $\sqsubseteq_i$, and $\preceq_i$ are quasiorderings.*

6.1 Closure under substitution

To show that $\preceq_i$ in fact preserves solvability (in the sense explained above), we first show that $\preceq_i$ is closed under substitution. This is done by induction on $i \leq k$. The base case, $i = k$, amounts to showing that $\preceq$ is closed under substitution. This requires some notation.

For a set of terms T, we define $\mathsf{an}_c(T)$ to be the closure of T under decomposition, $\mathsf{an}_e(T)$ to be the set of messages obtained by decrypting messages in T using keys already present in T, and $\mathsf{an}_{cec}(T)$ to be the iterated application of an_c, an_e, and an_c on T (in this order). Formally,

$$\begin{aligned}
\mathsf{an}_c(T) &:= \{y \in \mathcal{M} \mid \exists x, z \in \mathcal{M}_\varepsilon : xyz \in T\},\\
\mathsf{an}_e(T) &:= \{x \in \mathcal{M} \mid \exists a \in T \cap \mathcal{N} :\\
&\qquad\qquad \mathsf{enc}_a(x) \in T\} \cup T,\\
\mathsf{an}_{cec}(T) &:= \mathsf{an}_c(\mathsf{an}_e(\mathsf{an}_c(T))).
\end{aligned}$$

It is easy to see that

$$\mathsf{an}(T) = \bigcup_{i \geq 0} \mathsf{an}^i_{cec}(T), \tag{1}$$

where $\mathsf{an}^0_{cec}(T) := T$ and

$$\mathsf{an}^{i+1}_{cec}(T) := \mathsf{an}_{cec}(\mathsf{an}^i_{cec}(T)).$$

We write $\mathsf{an}^i_{cec}(t_0, \dots, t_{n-1}, T)$ for $\mathsf{an}^i_{cec}(\{t_0, \dots, t_{n-1}\} \cup T)$ and $\mathsf{an}^i_{cec,\mathcal{N}}(t_0, \dots, t_{n-1}, T)$ for $\mathsf{an}^i_{cec}(t_0, \dots, t_{n-1}, T) \cap \mathcal{N}$.

Given a term t and $N \subseteq \mathcal{N}$, we say that a subterm t' of t is *N-accessible*[5] in t if

1. $t' \in \mathsf{an}_c(t)$ or
2. there exists $\mathsf{enc}_a(t'') \in \mathsf{an}_c(t)$, $a \in N$, and t' is N-accessible in t''.

Lemma 4. *Let $x_0, x_0', \dots, x_{n-1}, x_{n-1}' \in \mathcal{M}$ be messages, $t(v_0, \dots, v_{n-1})$ be a term, $m := t[x_0, \dots, x_{n-1}]$, and $m' := t[x_0', \dots, x_{n-1}']$. If $x_i \preceq x_i'$ for all $i < n$, then $m \preceq m'$.*

Proof. Because of (1) it suffices to show

$$N_i := \mathsf{an}^i_{cec,\mathcal{N}}(m, N) \subseteq \mathsf{an}_{\mathcal{N}}(m', N) =: N_{m'}$$

for every $i \geq 0$ and $N \subseteq \mathcal{N}$.

Let $i = 0$ and $a \in N_0$. If $a \in N$, there is nothing to show. Otherwise, $t = a$ or there exists $i < n$ with $x_i = a$ (i.e., $t = v_i$). In the former case it immediately follows that $a \in N_{m'}$. In the latter case, $x_i \preceq x_i'$ implies $a \in \mathsf{an}_{\mathcal{N}}(x_i', N)$, and thus $a \in N_{m'}$.

[5] This notion was also defined in [3].

Let $i \geq 0$ and $a \in N_{i+1}$. If $a \in N_i$, the induction hypothesis yields $a \in N_{m'}$. Otherwise, there must exist $x \in \mathsf{an}^i_{cec}(m, N)$, a message z, and $b \in N_i$ with $x = \mathsf{enc}_b(z)$ and $a \in \mathsf{an}_c(z)$. We distinguish two cases.

(i) There exists $t'(v_0, \ldots, v_{n-1})$ such that $\mathsf{enc}_b(t')$ is a subterm of t, $z = t'[x_0, \ldots, x_{n-1}]$, and t' is N_i-accessible in t. Thus, by the induction hypothesis, t' is $N_{m'}$-accessible in t. Consequently, if $a \in \mathsf{an}_c(t')$, then $a \in N_{m'}$. Otherwise, there must exist $i < n$, and terms t_0, t_1 such that $t' = t_0 v_i t_1$, and $a \in \mathsf{an}_c(\{x_i\})$. Now $x_i \preceq x'_i$ implies $a \in \mathsf{an}(x'_i)$, and thus $a \in N_{m'}$.

(ii) There exists $i < n$ such that x, and thus z, is a submessage of x_i. By the induction hypothesis $N_i \subseteq N_{m'}$. We know that z and x_i are N_i-accessible in m. Thus, x'_i is N_i-accessible in m', and therefore also $N_{m'}$-accessible. Now $x_i \preceq x'_i$ implies $\mathsf{an}(x_i, N_i) \subseteq \mathsf{an}(x'_i, N_i)$, and thus $a \in \mathsf{an}(x_i, N_i)$ yields $a \in \mathsf{an}(x'_i, N_i)$. Consequently, $a \in \mathsf{an}(x'_i, N_{m'})$. Finally, since x'_i is $N_{m'}$-accessible in m', we obtain $a \in N_{m'}$. □

We generalize Lemma 4 to $\preceq_i$ and $\sqsubseteq_i$.

Lemma 5. *Let $x_0, x'_0, \ldots, x_{n-1}, x'_{n-1} \in \mathcal{M}_\varepsilon$, $i \leq k$, and $t(v_0, \ldots, v_{n-1})$ be a linear term. Let $m := t[x_0, \ldots, x_{n-1}]$ and $m' := t[x'_0, \ldots, x'_{n-1}]$. Then, $x_j \preceq_i x'_j$ ($x_j \sqsubseteq_i x'_j$) for all $j < n$ implies $m \preceq_i m'$ ($m \sqsubseteq_i m'$).*

Proof. W.l.o.g., we can assume that $x_j \neq \varepsilon$ and $x'_j \neq \varepsilon$ for all $j < n$, since otherwise $x_j = x'_j = \varepsilon$, and we can remove v_j from t altogether.

We prove the claim simultaneously for $\sqsubseteq_i$ and $\preceq_i$ by induction on i. If $k = i$, the statement for $\preceq_i$ follows immediately from Lemma 4. For $\sqsubseteq_i$ there is nothing to show. Let $i < k$.

Claim I. From $x_j \sqsubseteq_i x'_j$ for all $j < n$ it follows $m \sqsubseteq_i m'$.

Proof of Claim I. Let $p, q \in Q_i$, $y \in \mathcal{M}_\varepsilon$ with $\pi := p(m, y)q \in_s A_i$. If v_j occurs in t, then π contains a subpath of the form $p_j(x_j, y_j)p'_j \in_s \mathcal{A}_i$. The definition of message transducer guarantees that $y_j \in \mathcal{M}_\varepsilon$. Moreover, there exists a linear term $t'(v_0, \ldots, v_{n-1})$ such that $y = t'[v_0/y_0, \ldots, v_{n-1}/y_{n-1}]$. Because $x_j \sqsubseteq_i x'_j$, there exists $y'_j \in \mathcal{M}_\varepsilon$ with $p_j(x'_j, y'_j), p'_j \in_s \mathcal{A}_i$ and $y_j \preceq_{i+1} y'_j$. Define $y' := t'[v_0/y'_0, \ldots, v_{n-1}/y'_{n-1}]$. By the induction hypothesis, $y \preceq_{i+1} y'$. Finally, replacing the subpaths $p_j(x_j, y_j)p'_j$ in π by $p_j(x'_j, y'_j), p'_j$ shows that $p(m', y')q \in_s \mathcal{A}_i$. Thus, $m \sqsubseteq_i m'$.

Claim II. From $x_j \preceq_i x'_j$ for all $j < n$ it follows that $m \preceq_i m'$.

Proof of Claim II. From $x_j \preceq_i x'_j$, $j < n$, it follows that $x_j \preceq_{i+1} x'_j$. Thus, by the induction hypothesis, $m \preceq_{i+1} m'$. Since $x_j \preceq_i x'_j$ implies $x_j \sqsubseteq_i x'_j$, Claim I yields $m \sqsubseteq_i m'$.

Let $N \subseteq \mathcal{N}$ and $x \in \mathsf{an}(m, N)$ with $x = \mathsf{enc}_a(z)$ for some message z and $a \in \mathcal{N}$. Let $N_m := \mathsf{an}_\mathcal{N}(m, N)$ and $N_{m'} := \mathsf{an}_\mathcal{N}(m', N)$. Lemma 4 implies $N_m \subseteq N_{m'}$. Note that since x is of the form $\mathsf{enc}_a(\cdot)$, it cannot happen that just part of some x_i belongs to x, and therefore it suffices to consider the two following cases.

(i) There exists a subterm $t'(v_0, \ldots, v_{n-1})$ of t such that $x = t'[x_0, \ldots, x_{n-1}]$ and $t' \notin \{v_0, \ldots, v_{n-1}\}$. Let $x' := t'[x'_0, \ldots, x'_{n-1}]$. We know that t' is N_m-accessible in t. Thus, t' is also $N_{m'}$-accessible in t. In particular, $x' \in \mathsf{an}(m', N)$. Since t' is not a variable, it follows that t' is of the form $\mathsf{enc}_a(t'')$ for some term $t''(v_0, \ldots, v_{n-1})$. Thus, x' has the form $\mathsf{enc}_a(z')$ for some message z'. Finally, Claim I implies $x \sqsubseteq_i x'$.

(ii) There exists $j < n$ such that x is a subterm of x_j. In particular, x_j is N_m-accessible in m. Thus, x'_j is $N_{m'}$-accessible in m'. We know $x \in \mathsf{an}(x_j, N_m)$. Then, $x_j \preceq_i x'_j$ implies that there exists $x' \in \mathsf{an}(x'_j, N_m)$ of the form $\mathsf{enc}_b(z')$ for some message z' and $b \in \mathcal{N}$ with $x \sqsubseteq_i x'$. In particular, $x' \in \mathsf{an}(x'_j, N_{m'})$, and given that x'_j is $N_{m'}$-accessible in m', we conclude $x' \in \mathsf{an}(m', N)$. □

6.2 The ordering $\preceq_i$ is solvability preserving

We show that $\preceq_i$ is solvability preserving by induction on $i \leq k$. The base case, $i = k$, is a consequence of the following lemma.

Lemma 6. *For all $m, m' \in \mathcal{M}$, $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$, if $m \preceq m'$, then $\mathsf{an}_\mathcal{N}(m, \mathcal{K}) \subseteq \mathsf{an}_\mathcal{N}(m', \mathcal{K})$.*

Proof. We show

$$N_i := \mathsf{an}^i_{cec,\mathcal{N}}(m, \mathcal{K}) \subseteq \mathsf{an}_\mathcal{N}(m', \mathcal{K}) =: N_{m'}$$

for every $i \geq 0$ by induction on i.

Let $i = 0$ and $a \in N_0$. If $a \in \mathcal{K}$, nothing is to show. Otherwise, $m = a$, and $m \preceq m'$ implies $\mathsf{an}_\mathcal{N}(m) \subseteq \mathsf{an}_\mathcal{N}(m')$, and thus $a \in N_{m'}$.

Let $i \geq 0$ and $a \in N_{i+1}$. If $a \in N_i$, the induction hypothesis yields $a \in N_{m'}$. Otherwise, there must exist $x \in \mathsf{an}^i_{cec}(m, \mathcal{K})$, a message z, and $b \in N_i$ with $x = \mathsf{enc}_b(z)$ and $a \in \mathsf{an}_c(z)$. We distinguish two cases.

(i) The messages x and z are submessages of some message x' in $\mathcal{K}$. In particular, x and z are N_i-accessible in x', and thus x and z are $N_{m'}$-accessible in x'. Consequently, $a \in N_{m'}$.

(ii) The messages x and z are submessages of m. We know $a \in \mathsf{an}_\mathcal{N}(m, N_i)$ and $\mathsf{an}_\mathcal{N}(m, N_i) \subseteq \mathsf{an}_\mathcal{N}(m', N_i)$. Now $N_i \subseteq N_{m'}$ implies $a \in \mathsf{an}_\mathcal{N}(m', N_{m'}) = N_{m'}$. □

Using Lemmas 5 and 6 we prove the main statement of this section.

Proposition 1. *Let $(\mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$, $i \leq k$, be a solvable instance of* PathProblem *and $m \in \mathcal{K}$. Then, for every $\overline{m} \in \mathcal{M}_\varepsilon$ with $m \preceq_i \overline{m}$, the instance $(\overline{\mathcal{K}}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$ with $\overline{\mathcal{K}} := \mathcal{K} \setminus \{m\} \cup \{\overline{m}\}$ is also solvable.*

Proof. The proof is by induction on $i \leq k$. If $m = \varepsilon$, then, by definition of $\preceq_i$, $\overline{m} = \varepsilon$ and there is nothing to show. Therefore, assume $m \neq \varepsilon$, and thus $\overline{m} \neq \varepsilon$. The induction basis, $i = k$, immediately follows from Lemma 6.

Let $i < k$. Define $N := \mathsf{an}_\mathcal{N}(\mathcal{K})$ and $M := \{\mathsf{enc}_a(z) \in \mathsf{an}(m, N) \mid z \in \mathcal{M} \text{ and } a \in \mathcal{N}\}$. Let $(m_i, m'_i, \ldots, m_{k-1},$

m'_{k-1}) be a solution of $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$. Thus, $m_i \in \mathsf{d}(\mathcal{K})$. Since $\mathsf{d}(\mathcal{K}) = \mathsf{syn}(\mathsf{an}(\mathcal{K}))$, the derivation of m_i can be divided into an analysis phase and a synthesis phase. We may assume that in the analysis phase messages are decomposed as far as possible and that to construct m_i only those messages are used that in the analysis phase could not further be decomposed. Let $\{x_0, \dots, x_{n-1}\}$ be the multiset of the messages in M that have been used to construct m_i where an $x_j \in M$ occurs in this multiset as many times as it was used in the construction of m_i. Then, there exists a linear term $t(v_0, \dots, v_{n-1})$ with $m_i = t[x_0, \dots, x_{n-1}]$. Since $m \preceq_i \overline{m}$, for every x_j, $j < n$, there exists $\overline{x}_j \in \mathsf{an}(\overline{m}, N)$ with $x_j \sqsubseteq_i \overline{x}_j$. Define $\overline{m}_i := t[\overline{x}_0, \dots, \overline{x}_{n-1}]$. Since, by Lemma 6, $N \subseteq \mathsf{an}(\overline{\mathcal{K}})$, we can conclude $\overline{x}_j \in \mathsf{an}(\overline{\mathcal{K}})$, and it follows $\overline{m}_i \in \mathsf{d}(\overline{\mathcal{K}})$, since to construct $\overline{m}_i$ one only has to replace x_j by $\overline{x}_j$ in the synthesis phase of the derivation of m_i.

We may assume that $x_j \neq \varepsilon$, because otherwise $x_j = \overline{x_j} = \varepsilon$, and we can remove v_j from t altogether. Using that $q_i^I(m_i, m'_i)q_i^F \in \mathcal{A}_i$, one shows, similar to the proof of Claim I in Lemma 5, that there exists a message $\overline{m}'_i$ with $q_i^I(\overline{m}_i, \overline{m}'_i)q_i^F \in \mathcal{A}_i$ and $m'_i \preceq_{i+1} \overline{m}'_i$. Thus, since $(\mathcal{K} \cup \{m'_i\}, \mathcal{A}_{i+1}, \dots, \mathcal{A}_{k-1})$ has a solution, by the induction hypothesis, the instance $(\mathcal{K} \cup \{\overline{m}'_i\}, \mathcal{A}_{i+1}, \dots, \mathcal{A}_{k-1})$ is also solvable. Finally, since $m \preceq_i \overline{m}$ implies that $m \preceq_{i+1} \overline{m}$, induction yields a solution for $(\overline{\mathcal{K}} \cup \{\overline{m}'_i\}, \mathcal{A}_{i+1}, \dots, A_{k-1})$, and adding $\overline{m}_i$ and $\overline{m}'_i$ to this solution solves the instance $(\overline{\mathcal{K}}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$. □

6.3 Index of the solvability preserving equivalence relation

If $\leq$ is a quasiordering on a set S, then the relation $\equiv$ with $a \equiv b$ iff $a \leq b$ and $b \leq a$ for all $a, b \in S$ is an equivalence relation on S. The *equivalence class* of a modulo $\equiv$ is denoted $[a]_\equiv := \{b \mid a \equiv b\}$. The *index* $I(\equiv)$ of $\equiv$ is the number of different equivalence classes over S. In this section we show for different equivalence relations that their index is finite. This is done as follows. We first define a mapping f from S into a *finite* set T. Then, we show that $f(a) = f(b)$ implies $a \equiv b$ for every $a, b \in S$. From this it immediately follows that $I(\equiv) \leq |T|$. We will call f an *index mapping* of $\equiv$. If $S' \subseteq S$ is a finite subset of S and $f(a) = f(b)$ implies $a \equiv b$ for every $a, b \in S \setminus S'$, then we call f an *S'-index mapping* of $\equiv$. Note that in this case we can conclude $I(\equiv) \leq |T| + |S'|$. If $S' = \{s\}$ we write s-index mapping instead of $\{s\}$-index mapping.

In what follows, let $\equiv$, $\equiv_i$, and $=_i$ denote the equivalence relations corresponding to $\preceq$, $\preceq_i$, and $\sqsubseteq_i$, respectively. We show that the index of these relations is finite. To this end, we first consider $\equiv$.

Clearly, $f_\equiv(m) := (\mathsf{an}_\mathcal{N}(m, N) | N \subseteq \mathcal{N})$ is an index mapping of $\equiv$. Thus:

Lemma 7.

$$I(\equiv) \leq 2^{|\mathcal{N}| \cdot 2^{|\mathcal{N}|}}.$$

The following proposition generalizes the lemma to $\equiv_i$ and $=_i$. Note that $=_i$ is only defined for $i < k$.

Proposition 2. *We have $I(\equiv_k) = I(\equiv) + 1$ and for every $i < k$:*

$$I(=_i) \leq 2^{I(\equiv_{i+1}) \cdot 2^{|Q_i|^2}} + 1,$$
$$I(\equiv_i) \leq I(\equiv_{i+1}) \cdot 2^{I(=_i) \cdot 2^{|\mathcal{N}|}} \cdot I(=_i) + 1.$$

Proof. Obviously, $I(\equiv_k) = I(\equiv) + 1$. (Note that ε is handled differently for $\equiv_k$ and $\equiv$.) Assume that $i < k$ and the claim holds for $\equiv_{i+1}$. We first bound the index of $=_i$ and then that of $\equiv_i$.

We introduce a new equivalence relation on tuples (x, y) with $x, y \in \mathcal{M}_\varepsilon$. For every $x, x', y, y' \in \mathcal{M}_\varepsilon$ define: $(x, y) =_i^t (x', y')$ iff

- $y \equiv_{i+1} y'$, and
- $p(x, y)q \in_s \mathcal{A}_i$ iff $p(x', y')q \in_s \mathcal{A}_i$, for every $p, q \in Q_i$.

It is easy to see that $f_{=_i^t}(x, y) := ([y]_{\equiv_{i+1}}, \{(p, q) \mid p(x, y)q \in_s \mathcal{A}_i\})$ is an index mapping of $=_i^t$. It follows that

$$I(=_i^t) \leq I(\equiv_{i+1}) \cdot 2^{|Q_i|^2}.$$

Using $=_i^t$, one easily concludes that $f_{=i}(x) := \{[(x, y)]_{=_i^t} \mid y \in \mathcal{M}_\varepsilon\}$ is an ε-index mapping of $=_i$. This yields:

$$I(=_i) \leq 2^{I(=_i^t)} + 1 \leq 2^{I(\equiv_{i+1}) \cdot 2^{|Q_i|^2}} + 1.$$

We now consider $I(\equiv_i)$. Define $M_{i,m,N} := \{[x]_{=_i} \mid x \in \mathsf{an}(m, N)$ and $x = \mathsf{enc}_a(z)$ for $z \in \mathcal{M}$ and $a \in \mathcal{N}\}$, and $f_{\equiv i}(m) := ([m]_{\equiv_{i+1}}, (M_{i,m,N} \mid N \subseteq \mathcal{N}), [m]_{=i})$. It is not hard to see that $f_{\equiv_i}$ is an ε-index mapping of $\equiv_i$. As an immediate consequence, we obtain that

$$I(\equiv_i) \leq I(\equiv_{i+1}) \cdot 2^{I(=_i) \cdot 2^{|\mathcal{N}|}} \cdot I(=_i) + 1.$$

□

7 Bounding the depth of input messages

We show the following proposition.

Proposition 3. *If $(m_i, m'_i, \dots, m_{k-1}, m'_{k-1})$ is a solution of the instance $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$ of* PathProblem*, then there exists a solution $(\overline{m}_i, \overline{m}'_i, \dots, \overline{m}_{k-1}, \overline{m}'_{k-1})$ of this instance with $\mathsf{depth}(\overline{m}_i) \leq I(=_i) + \mathsf{depth}(\mathcal{K}) + 1$.*

In the following section, it is shown that the depth of m'_i can be bounded. As an immediate consequence, we will obtain that to find solutions for the path problem it suffices to consider only messages of depth bounded in the size of the problem instance.

To prove Proposition 3, let $(m_i, m'_i, \dots, m_{k-1}, m'_{k-1})$ be a solution of the instance $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$ of the path problem and assume that $\mathsf{depth}(m_i) > I(=_i) + \mathsf{depth}(\mathcal{K}) + 1$.

Then, there exist linear terms t and t' each containing exactly one variable and messages x, x' such that t' and x' start with an encryption symbol, $m_i = t[t'[x']]$, $x = t'[x']$, $\mathsf{depth}(x') > \mathsf{depth}(\mathcal{K})$, and $x =_i x'$. Note that $\mathsf{depth}(x') < \mathsf{depth}(x)$. Define $\overline{m}_i := t[x']$.

Since $m_i \in \mathsf{d}(\mathcal{K})$ and x' is a subterm of m_i with $\mathsf{depth}(x') > \mathsf{depth}(\mathcal{K})$, it follows that $x' \in \mathsf{d}(\mathcal{K})$ and all terms in $\mathsf{d}(\mathcal{K})$ containing x' as a subterm have been obtained in the synthesis phase, i.e., by composition and encryption. From this it follows that $\overline{m}_i \in \mathsf{d}(\mathcal{K})$.

We know that there exist words $\overline{x}, y, \overline{y}, z, \overline{z} \in \Sigma^*_{\mathcal{N}}$ and states q, q' such that $m_i = yxz$, $m'_i = \overline{yxz}$ and $q_i^I(y, \overline{y})q(x, \overline{x})q'(z, \overline{z})q_i^F \in \mathcal{A}_i$, where $q(x, \overline{x})q' \in_s \mathcal{A}_i$. In particular, $\overline{x}$ is a message. Since $x =_i x'$, we conclude that there exists $\overline{x}'$ with $q(x', \overline{x}')q' \in_s \mathcal{A}_i$ and $\overline{x} \preceq_{i+1} \overline{x}'$. Define $\overline{m}'_i := \overline{yx'z}$. By Lemma 5, $m'_i \preceq_{i+1} \overline{m}'_i$. Proposition 1 implies that the problem $(\mathcal{K} \cup \{\overline{m}'_i\}, \mathcal{A}_{i+1}, \dots, \mathcal{A}_{k-1})$ has a solution, say, $(\overline{m}_{i+1}, \overline{m}'_{i+1}, \dots, \overline{m}_{k-1}, \overline{m}'_{k-1})$. Thus, $(\overline{m}_i, \overline{m}'_i, \dots, \overline{m}_{k-1}, \overline{m}'_{k-1})$ is a solution of $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$.

By iterating this argument, the proposition follows.

8 Bounding the depth of output messages

In this section we show that the depth of output messages of a message transducer can be bounded by the depth of input messages. Standard arguments for word transducers do not apply here since a message transducer reads and returns messages rather than arbitrary words.

We first need to introduce some notions. A word $\alpha \in \Sigma^*_{\mathcal{N}}$ is a *left half-message* if α is a prefix of a message, i.e., there exists a word $\gamma \in \Sigma^*_{\mathcal{N}}$ such that $\alpha\gamma$ is a message. In Sect. 9, we also consider *right half-messages*, i.e., suffices of messages. For a left half-message α, the *level* $l(\alpha)$ *of* α is defined as the number of symbols "$\mathsf{enc}_a($", for some $a \in \mathcal{N}$, without matching closing parentheses. Analogously, for a right half-message γ, the *level* $l(\gamma)$ *of* γ is the number of closing parenthesis in γ without a matching encryption symbol "$\mathsf{enc}_a($", for some $a \in \mathcal{N}$.

In what follows, let $\mathcal{A}$ be a message transducer and π be a path in $\mathcal{A}$ of the form

$$q_0(a_0, b_0)q_1(a_1, b_1) \cdots (a_{r-1}, b_{r-1})q_r \tag{2}$$

with $r > 0$ and $a_i, b_i \in \Sigma_{\mathcal{N}} \cup \{\varepsilon\}$ for every $i < r$ such that $a_0 \cdots a_{r-1}$, $b_0 \cdots b_{r-1} \in \mathcal{M}_\varepsilon$. We define $l_\pi(i) := l(a_0 \cdots a_{i-1})$ and $l'_\pi(i) := l(b_0 \cdots b_{i-1})$ for all $i \leq r$ to be the *input and output level function of* π, respectively. Furthermore, we define $l_m(i) := l(c_0 \cdots c_{i-1})$, $> i < s$, as the *level function of* m with $m = c_0 \cdots c_{s-1}$.

The following proposition says that at any position in π, the level of the output at this position is bounded by the level of the input, and thus the depth of the output message can be bounded by the depth of the input message. This is not obvious since message transducers can have transitions in which no input is read but output is produced and these transitions may form cycles. The proof uses the fact that when reading a message, a message transducer always outputs a message rather than an arbitrary word.

Proposition 4. *Let $\mathcal{A} = (Q, \Sigma_{\mathcal{N}}, \{q_I\}, \Delta, \{q_F\})$ be a message transducer, $n := |Q|$, and π be a path in $\mathcal{A}$ of the form of (2) such that $q_0 = q_I$ and $q_r = q_F$, or π is strict, i.e., $\pi \in_s \mathcal{A}$. Then, it follows that $l'_\pi(i) \leq (n^2 \cdot (2n+1)+1) \cdot (l_\pi(i)+1)$ for every $i \leq r$.*

We define

$$\mathsf{depth}(\mathcal{A}) := n^2 \cdot (2n+1) + 1.$$

As a corollary of the proposition, we obtain that the depth of the output of a message transducer is bounded by the depth of the input.

Corollary 1. *Let $\mathcal{A} = (Q, \Sigma_{\mathcal{N}}, I, \Delta, F)$ be a message transducer. Then, for every $m, m' \in \mathcal{M}_\varepsilon$ with $m' \in \mathcal{A}(m)$, or $p(m, m')q \in_s \mathcal{A}$, for $p, q \in Q$: $\mathsf{depth}(m') \leq \mathsf{depth}(\mathcal{A}) \cdot (\mathsf{depth}(m)+1)$.*

Together with Proposition 5, this yields:

Corollary 2. *If $(\mathcal{K}, \mathcal{A}_i, \dots, \mathcal{A}_{k-1})$ is a solvable instance of the* PathProblem*, then there exists a solution $(m_i, m'_i, \dots, m_{k-1}, m'_{k-1})$ of this instance such that the depth of m_i and m'_i is bounded in the size of the instance: $\mathsf{depth}(m_i) \leq I(=_i) + \mathsf{depth}(\mathcal{K}) + 1$ and $\mathsf{depth}(m'_i) \leq \mathsf{depth}(\mathcal{A}_i) \cdot (\mathsf{depth}(m_i)+1)$.*

By induction, we can deduce that the depth of all messages in a solution of an instance of the path problem can be bounded. Formally:

Corollary 3. *There exists a (computable) function* depth *such that for every solvable instance $\mathcal{I} = (\mathcal{K}, \mathcal{A}_0, \dots, \mathcal{A}_{k-1})$ of* PathProblem*, there exists a solution $(m_0, m'_0, \dots, m_{k-1}, m'_{k-1})$ with $\mathsf{depth}(m_i) \leq \mathsf{depth}(\mathcal{I})$ and $\mathsf{depth}(m'_i) \leq \mathsf{depth}(\mathcal{I})$ for every $i < k$.*

Proof of Proposition 4. First, we show that the length of paths in $\mathcal{A}$ can be restricted by $\mathsf{depth}(\mathcal{A})$. To do so, we cannot simply use the usual pumping argument on finite automata since if we truncate a path, we need to guarantee that the input label of the resulting path is still a message. Therefore, the path can only be cut at certain positions. One possibility is that the path is cut such that an input label of the form $\mathsf{enc}_a(\cdots \mathsf{enc}_b(\cdot) \cdots)$ is replaced by $\mathsf{enc}_b(\cdot)$. Alternatively, one can cut a path such that if the input label is of the form $xw\gamma$, it is replaced by $x\gamma$, where x and xw are left half-messages of the same level (and thus γ is a right half-message of this level). A little technical problem in this case is that $x\gamma$ may not be a message since it can contain a word of the form $\mathsf{enc}_a()$, which is not a message. The following lemma shows how to solve this problem.

Lemma 8. *Let $m = c_0 \cdots c_{r-1} \in \mathcal{M}$ be a message with $c_i \in \Sigma_{\mathcal{N}}$ for all $i < r$.*

1. *If for $0 \leq i < j \leq r$, $l_m(i) = l_m(j) = 0$, then $c_0 \cdots c_{i-1}\, c_j \cdots c_{r-1} \in \mathcal{M}_\varepsilon$.*
2. *If $0 < i < j < r$ with $l_m(i) = l_m(j)$, then $c_{i-1} \neq \mathsf{enc}_a($ for every $a \in \mathcal{N}$, or $c_j \neq)$ iff $c_0 \cdots c_{i-1} c_j \cdots\ c_{r-1} \in \mathcal{M}_\varepsilon$.*
3. *Given $0 < i_0 < i_1 < i_2 < r$ with $l_m(i_0) = l_m(i_1) = l_m(i_2)$, then $c_0 \cdots c_{i_0-1} c_{i_1} \cdots c_{r-1} \in \mathcal{M}_\varepsilon$ or $c_0 \cdots c_{i_1-1} c_{i_2} \cdots c_{r-1} \in \mathcal{M}_\varepsilon$.*

Proof. The first statement is obvious. To prove the second statement, observe that the condition $c_{i-1} \neq \mathsf{enc}_a($, for every $a \in \mathcal{N}$, or $c_j \neq)$ guarantees that $c_{i-1}c_j$ is not of the form $\mathsf{enc}_b()$, for some $b \in \mathcal{N}$. Having ruled out this possibility, it is easy to show that $c_0 \cdots c_{i-1} c_j \cdots c_{r-1}$ is a message. Conversely, if $c_0 \cdots c_{i-1} c_j \cdots c_{r-1}$ is a message, then, since it does not contain a submessage of the form $\mathsf{enc}_b()$, it follows that $c_{i-1} \neq \mathsf{enc}_a($, for every $a \in \mathcal{N}$, or $c_j \neq)$.

To prove statement 3, assume that neither $c_0 \cdots c_{i_0-1} c_{i_1} \cdots c_{r-1} \in \mathcal{M}_\varepsilon$ nor $c_0 \cdots c_{i_1-1} c_{i_2} \cdots\ c_{r-1} \in \mathcal{M}_\varepsilon$. From statement 2 it follows that $c_{i_0} = \mathsf{enc}_a($, for some $a \in \mathcal{N}$, $c_{i_1} =)$, $c_{i_1-1} = \mathsf{enc}_b($, for some $b \in \mathcal{N}$, and $c_{i_2} =)$. But then c contains as the submessage $c_{i_1-1}c_{i_1} = \mathsf{enc}_b()$, in contradiction to the fact that c is a message. □

Now we show how to bound the length of paths by $\mathsf{depth}(\mathcal{A})$.

Lemma 9. *Let π be a path of the form given by (2) in a message transducer $\mathcal{A}$, where the input label is a message (not necessarily the output label), and let n be the number of states of $\mathcal{A}$. Then, there exists a path π' from q_0 to q_r in $\mathcal{A}$ such that the length of π' is $< \mathsf{depth}(\mathcal{A})$ and the input label of π' is a message. Moreover, if $a_{r-1} \neq \varepsilon$, then the input label of the last transition in π' is distinct from ε.*

Proof. Let $m := a_0 \cdots a_{r-1}$. We first show that we can restrict the depth of an input label of a path from q_0 to q_r by n^2.

Assume that $\mathsf{depth}(m) > n^2$. Then, there exist $i_0 < j_0 < j_1 < i_1 \leq r$ such that $q_{i_0} = q_{j_0}$, $q_{j_1} = q_{i_1}$, $a_{i_0} = \mathsf{enc}_a($, for some $a \in \mathcal{N}$, $a_{i_1-1} =)$ (the corresponding closing parenthesis to a_{i_0}), and, analogously, $a_{j_0} = \mathsf{enc}_b($, for some $b \in \mathcal{N}$, and $a_{j_1-1} =)$. It follows that the path π' given as

$$q_0(a_0, b_0)q_1 \cdots q_{i_0}(a_{j_0}, b_{j_0})q_{j_0+1} \cdots$$
$$(a_{j_1-1}, b_{j_1-1})q_{j_1}(a_{i_1}, b_{i_1})q_{i_1+1} \cdots q_r$$

is also a path in $\mathcal{A}$ from q_0 to q_r such that its input label is a message. Note that the input label of the last transition of π and the one of π' coincide. Iterating this argument, we obtain a path from q_0 to q_r such that the input label is a message of depth $\leq n^2$.

Thus, from now on we may assume that $\mathsf{depth}(m) \leq n^2$. In particular, $l_\pi(i) \leq n^2$ for every $i \leq r$. Now assume $r \geq \mathsf{depth}(\mathcal{A})$. Then, there must exist an $l \leq n^2$ such that $l_\pi(i) = l$ for $> 2n+1$ many $i \leq r$. Thus, there exist $0 \leq i_0 < i_1 < i_2 < r$ such that $q_{i_0} = q_{i_1} = q_{i_2}$ and $l_\pi(i_0) = l_\pi(i_1) = l_\pi(i_2) = l$. It follows that for $j \in \{0, 1\}$ the path π'_j given as

$$q_0(a_0, b_0)q_1 \cdots q_{i_j}(a_{i_{j+1}}, b_{i_{j+1}})q_{i_{j+1}+1}$$
$$(a_{i_{j+1}+1}, b_{i_{j+1}+1})q_{i_{j+1}+2} \cdots q_r$$

is a path in $\mathcal{A}$ from q_0 to q_r. Lemma 8 implies that the input label of π'_0 or π'_1 is a message. Finally, since $i_2 < r$, the last transition in π'_j, for $j \in \{0, 1\}$, coincides with the one for π.

Iterating this construction, we obtain a path π' with the desired properties. □

Now we can prove Proposition 4. Let π be a path as stipulated in the proposition, and assume that there exists $i \leq r$ such that $l'_\pi(i) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(i) + 1)$. We may assume that $l_\pi(i)$ is minimal, i.e., there exists no $j \leq r$ such that $l_\pi(j) < l_\pi(i)$ and $l'_\pi(j) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(j) + 1)$. We distinguish two cases.

First, assume $l_\pi(i) > 0$. Let $j > i$ be minimal with $l_\pi(j) = l_\pi(i) - 1$. From the minimality of $l_\pi(i)$ it follows that $l'_\pi(j) \leq \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(j) + 1)$. But then there must exist more than $\mathsf{depth}(\mathcal{A})$ many positions s with $i \leq s \leq j - 1$ and $b_s =)$. [Otherwise, $l'_\pi(j) = l'_\pi(i) - g$ for some $g \leq \mathsf{depth}(\mathcal{A})$]. Consequently, $l'_\pi(j) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(i) + 1) - \mathsf{depth}(\mathcal{A}) = \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(j) + 1)$, in contradiction to the minimality of $l_\pi(i)$.)

By the choice of j we know that the word $a_i \cdots a_{j-2}$ is a message and that it is the input label of the subpath π' of π from q_i to q_{j-1}. [Note that $a_{j-1} =)$.] Lemma 9 implies that there is also a path π'' in $\mathcal{A}$ from q_i to q_{j-1} of length $< \mathsf{depth}(\mathcal{A})$ such that the input label of π'' is a message. Replacing π' in π by π'' yields a new, shorter path, say, $\overline{\pi}$, from q_0 to q_r, with the properties required in the proposition. In particular, the input label of $\overline{\pi}$ is a message. Let $l_{\overline{\pi}}(j)$ and $l'_{\overline{\pi}}(j)$ be the input and output level functions of $\overline{\pi}$, respectively. Moreover, let $\overline{j}$ denote the position in $\overline{\pi}$ corresponding to j in π. Then, we have $l'_{\overline{\pi}}(\overline{j}) > \mathsf{depth}(\mathcal{A}) \cdot (l_{\overline{\pi}}(\overline{j}) + 1)$ and $l_{\overline{\pi}}(\overline{j}) < l_{\overline{\pi}}(\overline{i}) = l_\pi(i)$ since $< \mathsf{depth}(\mathcal{A})$ parentheses have been closed between position i and $\overline{j} - 1$ in $\overline{\pi}$, i.e., $\leq \mathsf{depth}(\mathcal{A})$ parentheses between i and $\overline{j}$. Iterating this argument shows that there exists a path π with the desired properties such that $l_\pi(i)$, as chosen above, is 0. Thus, the case $l_\pi(i) = 0$ applies. (However, we will see that this case leads to a contradiction.)

Now assume $l_\pi(i) = 0$. Let π' be the path from q_i to q_r with input label $a_i \cdots a_{r-1}$ and output label $b_i \cdots b_{r-1}$. Note that $a_i \cdots a_{r-1}$ is a message. Since we have that $l'_\pi(i) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(i) + 1)$, the output label must contain $> \mathsf{depth}(\mathcal{A})$ closing parentheses. However, according to Lemma 9, there exists a path π'' from q_i to q_r in $\mathcal{A}$ such that the input label of π'' is a message and the length of π'' is $< \mathsf{depth}(\mathcal{A})$. Then, replacing π' in π by π'' yields a new path $\overline{\pi}$ from q_0 to q_r such that the input label is a message. Moreover, if π is strict, then the new path is

also strict: if the last transition in π' ended with an input label distinct from ε, then, according to Lemma 9, this can be achieved for π'' as well. Now, since the input label of $\overline{\pi}$ is a message, the conditions on message-transducers imply that the output label must also be a message. However, this is not true, since closing parentheses are missing.

9 The path truncation ordering

We extend $\preceq_i$ to the path truncation ordering $\preceq_i^l$ on half-messages and show that $\preceq_i^l$ is compatible with right concatenation of right half-messages (Sect. 9.1). In Sect. 9.2 we prove that the index of the equivalence relation corresponding to $\preceq_i^l$ is finite.

For the definition of left and right half-messages and their levels $l(\cdot)$, see Sect. 8. If α is a left half-message, then there exist unique messages $x_0, \dots, x_{l(\alpha)} \in \mathcal{M}_\varepsilon$ and $a_1, \dots, a_{l(\alpha)-1} \in \mathcal{N}$ such that

$$\alpha = x_{l(\alpha)}\mathsf{enc}_{a_{l(\alpha)}}(x_{l(\alpha)-1}\mathsf{enc}_{a_{l(\alpha)-1}}(x_{l(\alpha)-2} \cdots \mathsf{enc}_{a_1}(x_0. \tag{3}$$

We define the *j-level half-message of α* to be

$$\alpha_j := \mathsf{enc}_{a_j}(x_{j-1}\mathsf{enc}_{a_{j-1}}(x_{j-2} \cdots \mathsf{enc}_{a_1}(x_0 \tag{4}$$

for $1 \le j \le l(\alpha)$. Note that $l(\alpha_j) = j$. For convenience of notation, define $\alpha_0 := x_{l(\alpha)}$ to be the *0-level half message of α*. Note that α_0 is a message, i.e., $l(\alpha_0) = 0$, and that if α is a message, then $\alpha = \alpha_0$. Moreover, we define α^* to be the message obtained from α by adding the missing closing parentheses, i.e.,

$$\alpha^* := \alpha \underbrace{)\cdots)}_{l(\alpha)}.$$

Finally, let $p(\alpha) := a_{l(\alpha)} \cdots a_1$. In order to define $\preceq_i^l$, we first introduce the ordering $\preceq^l$.

Definition 8. *Let $l \ge 0$ and α, α' be nonempty left half-messages of level l, i.e., $\alpha \ne \varepsilon$, $\alpha' \ne \varepsilon$, and $l(\alpha) = l(\alpha') = l$. Define $\alpha \preceq^l \alpha'$ iff $\alpha^* \preceq \alpha'^*$ and $p(\alpha) = p(\alpha')$.*

Clearly, $\preceq^l$ is a quasiordering. For the definition of $\preceq_i^l$ we need some more notation. If α and β are left half-messages, and $p, q \in Q_i$, then $p(\alpha, \beta)q \in_h \mathcal{A}_i$ means that (i) $p(\alpha, \beta)q \in_s \mathcal{A}_i$, and (ii) there exist right half-messages γ and γ' and a state $q' \in Q_i$ such that $l(\gamma) = l(\alpha)$, $l(\gamma') = l(\beta)$, and $p(\alpha, \beta)q(\gamma, \gamma')q'$ is a strict path in $\mathcal{A}_i$. In other words, the strict path $p(\alpha, \beta)q$ can be extended to a strict path such that the input and output labels are messages.

The definition of the path truncation ordering is quite similar to the solvability preserving ordering. The main difference is that additional restrictions are imposed on j-level half-messages α_j.

Definition 9. *For every $l \ge 0$ and $i \le k$, the* path truncation ordering *$\preceq_i^l$ is defined as follows. For left half-messages α, α' of level l: $\alpha \preceq_i^l \alpha'$ iff (i) $\alpha = \alpha' = \varepsilon$ or (ii) $\alpha \ne \varepsilon$, $\alpha' \ne \varepsilon$, $\alpha \preceq^l \alpha'$, and if $i < k$, then*

1. *$\alpha \preceq_{i+1}^l \alpha'$,*
2. *$\alpha^* \preceq_i \alpha'^*$,*
3. *$\alpha_j \sqsubseteq_i^j \alpha'_j$ for every $j \le l$ with α_j and α'_j the j-level half messages of α and α', respectively.*

For $i < k$, $l \ge 0$, left half-messages α, α' of level l, we define: $\alpha \sqsubseteq_i^l \alpha'$ iff i) $\alpha = \alpha' = \varepsilon$, or ii) $\alpha \ne \varepsilon$, $\alpha' \ne \varepsilon$, and for every left half-message β and every $p, q \in Q_i$, $p(\alpha, \beta)q \in_h \mathcal{A}_i$ implies that there exists a left half-message β' with $l(\beta') = l(\beta)$ such that $p(\alpha', \beta')q \in_h \mathcal{A}_i$ and $\beta \preceq_{i+1}^{l(\beta)} \beta'$.

9.1 Right concatenation of right half-messages

We now show, by induction on i, that $\preceq_i^l$ is compatible with right concatenation of right half-messages. The case $i = k$ follows from:

Lemma 10. *Let α, α' be left half-messages of level $l \ge 0$. Then, $\alpha \preceq^l \alpha'$ implies $\alpha\gamma \preceq \alpha'\gamma$ for every right half-message γ of level l.*

Proof. Assume $\alpha \preceq^l \alpha'$. If $l = 0$, then α, α', and γ are messages. Thus, with $t := v_0 v_1$, Lemma 4 implies $\alpha\gamma = t[\alpha, \gamma] \preceq t[\alpha', \gamma] = \alpha'\gamma$. In what follows we assume $l > 0$.

We show $\mathsf{an}_\mathcal{N}(\alpha\gamma, N) \subseteq \mathsf{an}_\mathcal{N}(\alpha'\gamma, N)$ for every $N \subseteq \mathcal{N}$. To this end, we define two mappings F and F' from $2^\mathcal{N}$ into $2^\mathcal{N}$, where $2^\mathcal{N}$ denotes the powerset of $\mathcal{N}$. For every $N \subseteq \mathcal{N}$,

$$F(N) := \mathsf{an}_\mathcal{N}(\mathsf{enc}_{p(\alpha)}(\gamma), \mathsf{an}_\mathcal{N}(\alpha^*, N)) \text{ and}$$
$$F'(N) := \mathsf{an}_\mathcal{N}(\mathsf{enc}_{p(\alpha')}(\gamma), \mathsf{an}_\mathcal{N}(\alpha'^*, N)),$$

where $\mathsf{enc}_w(\gamma)$ for some nonempty word $w = a_0 \cdots a_{l-1} \in \mathcal{N}^+$ denotes the message

$$\mathsf{enc}_{a_0}(\mathsf{enc}_{a_1}(\cdots \mathsf{enc}_{a_{l-1}}(\gamma.$$

(Note that the corresponding closing parentheses to "$\mathsf{enc}_{a_j}($" are contained in γ).

Because $\alpha \preceq^l \alpha'$, we know that $p(\alpha) = p(\alpha')$ and $\alpha^* \preceq \alpha'^*$. Thus, $\mathsf{an}_\mathcal{N}(\alpha^*, N) \subseteq \mathsf{an}_\mathcal{N}(\alpha'^*, N)$. Consequently, $F(N) \subseteq F'(N)$. It is easy to verify that

$$\mathsf{an}_\mathcal{N}(\alpha\gamma, N) = \bigcup_{j \ge 0} F^j(N), \text{ and}$$
$$\mathsf{an}_\mathcal{N}(\alpha'\gamma, N) = \bigcup_{j \ge 0} F'^j(N).$$

From this the lemma follows. □

We can now prove the main proposition of this section.

Proposition 5. *Let α, α' be left half-messages of level $l \ge 0$ and let $i \le k$. Then, $\alpha \preceq_i^l \alpha'$ implies $\alpha\gamma \preceq_i \alpha'\gamma$ for every right half-message γ of level l.*

PROOF. Assume $\alpha \preceq_i^l \alpha'$. If $\alpha = \varepsilon$, then $\alpha' = \varepsilon$, and thus $\alpha\gamma \preceq_i \alpha'\gamma$. Assume $\alpha \neq \varepsilon$ and $\alpha' \neq \varepsilon$. The rest of the proof is by induction on $i \leq k$. The base case, $i = k$, follows from Lemma 10.

Now assume $i < k$. If $l = 0$, then α, α', and γ are messages, and $\alpha\gamma \preceq_i \alpha'\gamma$ follows from Lemma 5 with $t = v_0 v_1$. Therefore, we may assume that $l > 0$. To prove $\alpha\gamma \preceq_i \alpha'\gamma$, we show that the conditions in Definition 7 are met.

Denote by α_j and α'_j the j-level half-messages of α and α'. Let $p(\alpha) = a_l \cdots a_1$. Note that $p(\alpha') = p(\alpha)$.

First, $\alpha \preceq_i^l \alpha'$ implies $\alpha \preceq_{i+1}^l \alpha'$, and thus, with the induction hypothesis, we obtain $\alpha\gamma \preceq_{i+1} \alpha'\gamma$.

Second, let $N \subseteq \mathcal{N}$ and $x \in \mathsf{an}(\alpha\gamma, N)$ with $x = \mathsf{enc}_a(z)$ for some message z and $a \in \mathcal{N}$. We distinguish three cases:

1. x is a submessage of γ. Due to Lemma 10, $\alpha\gamma \preceq \alpha'\gamma$. Consequently, $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$. Using $p(\alpha) = p(\alpha')$, it follows that $x \in \mathsf{an}(\alpha'\gamma, N)$, and we can simply choose $x' := x$.
2. x is a submessage of α. It follows that $x \in \mathsf{an}(\alpha^*, \mathsf{an}_{\mathcal{N}}(\alpha\gamma, N))$. Because $\alpha^* \preceq_i \alpha'^*$, there exists $x' \in \mathsf{an}(\alpha'^*, \mathsf{an}(\alpha\gamma, N))$ such that x' is of the form $\mathsf{enc}_b(z')$ for some message z' and $b \in \mathcal{N}$ and $x \sqsubseteq_i x'$. Finally, since $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$ (Lemma 10), $x' \in \mathsf{an}(\alpha'^*, \mathsf{an}(\alpha'\gamma, N))$, and thus $x' \in \mathsf{an}(\alpha'\gamma, N)$.
3. x is of the form $\alpha_j\gamma'$ for some $1 \leq j \leq l$ and a right half-message γ' such that γ' is a prefix of γ with $l(\gamma') = l(\alpha_j)$. Note that $a = a_j$. Define $x' := \alpha'_j\gamma'$. Obviously, x' is a message of the form $\mathsf{enc}_a(z')$ for some message z'. Since $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$ and $p(\alpha) = p(\alpha')$, it follows that $x' \in \mathsf{an}(\alpha'\gamma, N)$. We need to show $x \sqsubseteq_i x'$.

   Let $p, q \in Q_i$, $y \in \mathcal{M}_\varepsilon$ with $p(x, y)q \in_s \mathcal{A}_i$. There exists $p' \in Q_i$, a left half-message β, and a right half-message β' such that $y = \beta\beta'$, $l(\beta) = l(\beta')$, $p(\alpha_j, \beta)p' \in_s \mathcal{A}_i$, and $p'(\gamma', \beta')q \in \mathcal{A}_i$. We know that $p(\alpha_j, \beta)p'(\gamma', \beta')q$ is a strict path in $\mathcal{A}_i$ and that $x = \alpha_j\gamma'$ and $y = \beta\beta'$ are messages. Thus, $p(\alpha_j, \beta)p' \in_h \mathcal{A}_i$. From $\alpha \preceq_i^l \alpha'$ we obtain $\alpha_j \sqsubseteq_i^j \alpha'_j$, and hence there exists a left half-message β'' with $l(\beta) = l(\beta'')$, $p(\alpha'_j, \beta'')p' \in_h \mathcal{A}_i$, and $\beta \preceq_{i+1}^{l(\beta)} \beta''$. This yields that $p(\alpha'_j, \beta'')p'(\gamma', \beta')q$ is a strict path from p to q in $\mathcal{A}_i$ with input label x' and output label $y' := \beta''\beta'$. By the induction hypothesis, $y = \beta\beta' \preceq_{i+1} \beta''\beta' = y'$.

Finally, we show $\alpha\gamma \sqsubseteq_i \alpha'\gamma$. Let $p, q \in Q_i$ and $y \in \mathcal{M}_\varepsilon$ with $p(\alpha\gamma, y)q \in_s \mathcal{A}_i$. Since $l > 0$, α has the form $x\alpha_l$ for some message x, i.e., $x = \alpha_0$. We first assume $x \neq \varepsilon$. Thus, there exist words $y_0, y_1, \beta, \beta' \in \Sigma_{\mathcal{N}}^*$ and states p_0, p_1, p_2 such that $p(x, y_0)p_0 \in_s \mathcal{A}_i$, $p_0(\varepsilon, y_1)p_1 \in \mathcal{A}_i$, $p_1(\alpha_l, \beta)p_2 \in_s \mathcal{A}_i$, and $p_2(\gamma, \beta')q \in A_i$, where the input label of the last transition of the latter path is $\neq \varepsilon$.

Since the first path is strict and x is a message, by the definition of message transducers it follows $y_0 \in \mathcal{M}_\varepsilon$. Since the path from p_1 to q is strict and $\alpha_l\gamma$ is a message, we know that $\beta\beta'$ is a message. In particular, β is a left half-message and β' is a right half-message with $l(\beta) = l(\beta')$. Finally, since $y = y_0 y_1 \beta\beta' \in \mathcal{M}_\varepsilon$ and y_0, $\beta\beta' \in \mathcal{M}_\varepsilon$, we can conclude $y_1 \in \mathcal{M}_\varepsilon$.

Now $\alpha \preceq_i^l \alpha'$ implies $\alpha_l \sqsubseteq_i^l \alpha'_l$, and we know that $p_1(\alpha_l, \beta)p_2(\gamma, \beta')q$ is a strict path in $\mathcal{A}_i$, and $\alpha_l\gamma$ and $\beta\beta'$ are messages. Thus, $p_1(\alpha_l, \beta)p_2 \in_h \mathcal{A}_i$. As a result, there exists a left half-message β'' with $l(\beta'') = l(\beta)$, $p_1(\alpha'_l, \beta'')p_2 \in_s \mathcal{A}_i$, and $\beta \preceq_{i+1}^{l(\beta)} \beta''$. Moreover, if $\alpha' = x'\alpha'_l$, then $\alpha \preceq_i^l \alpha'$ implies $\alpha_0 = x \sqsubseteq_i x' = \alpha'_0$. Thus there exists $y'_0 \in \mathcal{M}_\varepsilon$ with $p(x', y'_0), p_0 \in_s \mathcal{A}_i$ and $y_0 \preceq_{i+1} y'_0$. Consequently, replacing in the path $p(\alpha\gamma, y)q$ the subpath $p(x, y_0)p_0$ by $p(x', y'_0)p_0$ and the subpath $p_1(\alpha_l, \beta)p_2$ by $p_1(\alpha'_l, \beta'')p_2$ yields a strict path from p to q with input label $\alpha'\gamma$ and output label $y' := y'_0 y_1 \beta''\beta' \in \mathcal{M}_\varepsilon$.

It remains to show $y \preceq_{i+1} y'$. By induction, $\beta \preceq_{i+1}^{l(\beta)} \beta''$ implies $\beta\beta' \preceq_{i+1} \beta''\beta'$. We also know $y_0 \preceq_{i+1} y'_0$ and $y_1 \preceq_{i+1} y_1$. Thus, by Lemma 5, with $t = v_0 v_1 v_2$ we obtain $y = t[y_0, y_1, \beta\beta'] \preceq_{i+1} t[y'_0, y_1, \beta''\beta'] = y'$.

If $x = \varepsilon$ and x' is defined as above, it follows that $x' = \varepsilon$, because $x \sqsubseteq_i x'$. The rest of the proof is similar to the case $x \neq \varepsilon$. □

9.2 Index of the path truncation ordering

Let $\equiv^l$, $\equiv_i^l$, and $=_i^l$ denote the equivalence relations corresponding to $\preceq^l$, $\preceq_i^l$, and $\sqsubseteq_i^l$, respectively. We show that these relations have finite index.

We start with $\equiv^l$. Obviously, $f_{\equiv^l}$ with $f_{\equiv^l}(\alpha) := ([\alpha^*]_\equiv, p(\alpha))$ for every left half-message α of level l is an index mapping (Sect. 6.3) for $\equiv^l$. It follows that:

Lemma 11. *For every $l \geq 0$:*

$$I(\equiv^l) \leq I(\equiv) \cdot |\mathcal{N}|^l.$$

The proof of the following proposition is very similar to the one for Proposition 2. However, it requires Proposition 4.

Proposition 6. *For every $l \geq 0$: $I(\equiv_k^l) = I(\equiv^l) + 1$ and*

$$I(=_i^l) \leq 2^{\left(I(\equiv_{i+1}^{\mathsf{depth}_l(\mathcal{A}_i)}) \cdot 2^{|Q_i|^2}\right) \cdot \mathsf{depth}_l(\mathcal{A}_i)} + 1$$
$$I(\equiv_i^l) \leq I(\equiv_{i+1}^l) \cdot I(\equiv_i) \cdot I(=_i^l)^l \cdot I(=_i) + 1$$

for every $i < k$ and $\mathsf{depth}_l(\mathcal{A}_i) := \mathsf{depth}(\mathcal{A}_i) \cdot (l + 1)$.

PROOF. For $\equiv_k^l$ the claim is obvious. Assume that $i < k$ and that the claim holds for $\equiv_{i+1}^l$. We first consider $=_i^l$.

For every $l, l' \geq 0$, we introduce a new equivalence relation on tuples of left half-messages: For left half-messages $\alpha, \alpha', \beta, \beta'$ with $l(\alpha) = l(\alpha') = l$ and $l(\beta) = l(\beta') = l'$ we define: $(\alpha, \beta) =_i^{l,l'} (\alpha', \beta')$ iff

- $\beta \equiv_{i+1}^{l'} \beta'$ and
- $p(\alpha, \beta)q \in_h \mathcal{A}_i$ iff $p(\alpha', \beta')q \in_h \mathcal{A}_i$ for every $p, q \in Q_i$.

Just as for $=_i^t$, in the proof of Proposition 2 one shows

$$I(=_i^{l,l'}) \leq I(\equiv_{i+1}^{l'}) \cdot 2^{|Q_i|^2}.$$

To show that $=_i^l$ has a finite index, define for $l, l' \geq 0$ and a left half-message α of level l the set

$$M_{i,\alpha}^{l,l'} := \{[(\alpha,\beta)]_{=_i^{l,l'}} \mid \beta \text{ left half-message of level } l'\}$$

and the tuple

$$M_{i,\alpha}^{l} := (M_{i,\alpha}^{l,l'} \mid l' \leq \mathsf{depth}_l(\mathcal{A}_i)).$$

Define $f_{=_i^l}(\alpha) := M_{i,\alpha}^l$ for every nonempty left half-message α.

Claim. $f_{=_i^l}$ is an ε-index mapping for $=_i^l$.

Proof of the claim. Obviously, the range of $f_{=_i^l}$ is finite. Assume $M_{i,\alpha}^l = M_{i,\alpha'}^l$ for nonempty left half-messages α, α' of level l. We show $\alpha \sqsubseteq_i^l \alpha'$; $\alpha' \sqsubseteq_i^l \alpha$ follows by symmetry. Let $p, q \in Q_i$ and β be a left half-message with $p(\alpha,\beta)q \in_h \mathcal{A}_i$. By definition of $\in_h$, there exist right half-messages γ and γ' and a state $q' \in Q_i$ such that $p(\alpha,\beta)q(\gamma,\gamma')q'$ is a strict path in $\mathcal{A}_i$. Then, Proposition 4 implies $l' := l(\beta) \leq \mathsf{depth}_l(\mathcal{A}_i)$. Now, using $M_{i,\alpha}^l = M_{i,\alpha'}^l$, it follows $[(\alpha,\beta)]_{=_i^{l,l'}} \in M_{i,\alpha}^{l,l'} = M_{i,\alpha'}^{l,l'}$. Thus, there exists a left half-message β' with $l(\beta') = l'$ with $[(\alpha,\beta)]_{=_i^{l,l'}} = [(\alpha',\beta')]_{=_i^{l,l'}}$. In particular, $\beta \equiv_{i+1}^{l'} \beta'$, and from $p(\alpha,\beta)q \in_h \mathcal{A}_i$ it follows that $p(\alpha',\beta')q \in_h \mathcal{A}_i$. This concludes the proof of the claim.

By induction on r simultaneously for $\equiv_i^r$ and $=_i^r$, it is easy to see that $I(\equiv_i^r) \leq I(\equiv_i^{r'})$ and $I(=_i^r) \leq I(=_i^{r'})$ for every $r \leq r'$. Now, from the claim it follows:

$$\begin{aligned} I(=_i^l) &\leq 2^{I\left(=_i^{l,\mathsf{depth}_l(\mathcal{A}_i)}\right)\cdot \mathsf{depth}_l(\mathcal{A}_i)} + 1 \\ &\leq 2^{\left(I\left(\equiv_{i+1}^{\mathsf{depth}_l(\mathcal{A}_i)}\right)\cdot 2^{|Q_i|^2}\right)\cdot \mathsf{depth}_l(\mathcal{A}_i)} + 1. \end{aligned}$$

Finally, let $f_{\equiv_i^l}(\alpha) := ([\alpha]_{\equiv_{i+1}^l}, [\alpha^*]_{\equiv i}, [\alpha_0]_{=_i^1}, \ldots, [\alpha_l]_{=_i^l})$ for every nonempty left half-message α of level l. It is easy to see that $f_{\equiv_i^l}$ is an ε-index mapping of $\equiv_i^l$. From this, the bound on $I(\equiv_i^l)$ claimed in the proposition follows immediately. □

10 Proof of Theorem 2

Putting everything together, we now show that the path problem is decidable. This will conclude the proof of Theorem 2.

We show that to find a solution of an instance of PathProblem, it suffices to consider paths (and thus messages) of length restricted in the size of the problem instance. To this end, we assume that the instance $(\mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$ has the solution $(m_i, m_i', \ldots, m_{k-1}, m_{k-1}')$ and then construct a solution with short paths. We do so by induction on $i \leq k$. For $i = k$, nothing is to be shown. For the induction step we need some notation.

Assume

$$\pi = q_0(a_0,b_0)q_1(a_1,b_1)\cdots(a_{r-1},b_{r-1})q_r$$

is a path in $\mathcal{A}_i$ with $a_0 \cdots a_{r-1} = m_i$, $b_0 \cdots b_{r-1} = m_i'$, $q_0 = q_i^I$, and $q_r = q_i^F$. Define $m_i(j,j') := a_j \cdots a_{j'-1}$ and $m_i'(j,j') := b_j \cdots b_{j'-1}$ for all $0 \leq j \leq j' \leq r$. We define $l_j := l_\pi(j)$ and $l_j' := l_\pi'(j)$, where l_π and l_π' are the input and output level functions of π (cf. Sect. 8).

Due to Corollary 3, we may assume that the depth of m_i and m_i' is bounded in the size of the problem instance.

Define $N := \mathsf{an}_{\mathcal{N}}(\mathcal{K})$ and $M := \{\mathsf{enc}_a(z) \in \mathsf{an}(\mathcal{K}) \mid z \in \mathcal{M} \text{ and } a \in \mathcal{N}\}$. Similar to the proof of Proposition 1, let n be the number of times a message in M was used to construct $m_i \in \mathsf{d}(\mathcal{K})$ from $\mathcal{K}$, and let $\{x_0, \ldots, x_{n-1}\}$ be the multiset of these messages; an $x_j \in M$ occurs in this multiset as many times as it was used in the derivation of m_i. Then, there exists a linear term $t(v_0, \ldots, v_{n-1}) \in \mathsf{d}(N \cup \{v_0, \ldots, v_{n-1}\})$ with $m_i = t[x_0, \ldots, x_{n-1}]$. Moreover, there exist positions $i_j, i_j' \leq r$ in π such that $x_j = a_{i_j} a_{i_j+1} \cdots a_{i_j'-1}$, $a_{i_j} \neq \varepsilon$, and $a_{i_j'-1} \neq \varepsilon$, for every $j < n$. That is, the subpath in π between the positions i_j and i_j' is a strict path in $\mathcal{A}_i$ with input label x_j.

Finally, we define a mapping f_π that indicates at which positions π can be truncated. It distinguishes between positions "inside" and "outside" an x_j. For every $j \leq r$, $f_\pi(j)$ is defined to be

$$(q_j, l_j, l_j', [m_i'(0,j)]_{\equiv_{i+1}^{l_j'}}, x_s, m_i(i_s, j))$$

if there exists $s < n$ such that $i_s < j < i_s'$, and

$$(q_j, l_j, l_j', [m_i'(0,j)]_{\equiv_{i+1}^{l_j'}})$$

otherwise. The following lemma shows how to truncate paths using f_π.

Lemma 12. *If there exist j_0, j_1, and j_2 with $0 \leq j_0 < j_1 < j_2 \leq r$ and $f_\pi(j_0) = f_\pi(j_1) = f_\pi(j_2)$, then there exists $u \in \{0,1\}$ and a solution $(\overline{m}_i, \overline{m}_i', \ldots, \overline{m}_{k-1}, \overline{m}_{k-1}')$ of $(\mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$ such that the path $\overline{\pi} :=$*

$$\begin{aligned} &q_0(a_0,b_0)q_1\cdots q_{j_u}(a_{j_u+1},b_{j_u+1})q_{j_u+1} \\ &(a_{j_u+1},b_{j_u+1})\cdots(a_{r-1},b_{r-1})q_r \end{aligned}$$

is a path in $\mathcal{A}_i$ from q_i^I to q_i^F with input label $\overline{m}_i = m_i(0,j_u)m_i(j_{u+1},r)$ and output label $\overline{m}_i' = m_i'(0,j_u)\, m_i'(j_{u+1},r)$.

Proof. From $f_\pi(j_0) = f_\pi(j_1) = f_\pi(j_2)$ it follows that $l_{j_0} = l_{j_1} = l_{j_2}$. Now, Lemma 8 implies that there exists $u \in \{0,1\}$ such that $\overline{m}_i := m_i(0,j_u)m_i(j_{u+1},r)$ is a message. Since $\overline{\pi}$ is a path from q_i^I to q_i^F with input label $\overline{m}_i$, from the properties of message transducers (cf. Definition 4) it follows that $\overline{m}_i' := m_i'(0,j_u)m_0'(j_{u+1},r)$ must be a message.

It remains to show that there exist messages $\overline{m}_{i+1}$, $\overline{m}'_{i+1}, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1}$ such that $(\overline{m}_i, \overline{m}'_i, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1})$ is a solution of $(\mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$.

We first show $\overline{m}_i \in \mathsf{d}(\mathcal{K})$. Note that, if $i_s < j_u < i'_s$ for some $s < n$, then $f_\pi(j_u) = f_\pi(j_{u+1})$ implies that there exists $s' < n$ with $x_s = x_{s'}$ and $m_i(i_s, j_u) = m_i(i_{s'}, j_{u+1})$. Thus, $x_s = m_i(i_s, j_u) m_i(j_{u+1}, i_{s'})$. That is, after removing the subpath in π between j_u and j_{u+1}, we still have x_s as a submessage. If there is no $s < n$ with $i_s < j_u < i'_s$, then in m_i only parts of t are removed, which when instantiated may or may not contain x_i's.

In either case, it is easy to conclude that there exists a linear term $\overline{t}(v_0, \ldots, v_{n-1}) \in \mathsf{d}(N \cup \{v_0, \ldots, v_{n-1}\})$ such that $\overline{m}_i := \overline{t}[x_0, \ldots, x_{n-1}]$. Thus, $\overline{m}_i \in \mathsf{d}(\mathcal{K})$.

Because $f_\pi(j_u) = f_\pi(j_{u+1})$, we know $[m'_i(0, j_u)]_{\equiv^{l'}_{i+1}} = [m'_i(0, j_{u+1})]_{\equiv^{l'}_{i+1}}$ with $l' := l'_{j_u} = l'_{j_{u+1}}$. Moreover, since we have $m'_i = m'_i(0, j_{u+1})\, m'_i(j_{u+1}, r)$ and $\overline{m}'_i = m'_i(0, j_u)\, m'_i(j_{u+1}, r)$, Proposition 5 implies $m'_i \equiv_{i+1} \overline{m}'_i$.

Now Proposition 1 guarantees that the instance $(\overline{\mathcal{K}}, \mathcal{A}_{i+1}, \ldots, A_{k-1})$ with $\overline{\mathcal{K}} := \mathcal{K} \cup \{\overline{m}'_i\}$ has a solution $(\overline{m}_{i+1}, \overline{m}'_{i+1}, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1})$. Thus, $(\overline{m}_i, \overline{m}'_i, \overline{m}_{i+1}, \overline{m}'_{i+1}, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1})$ solves $(\mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$. □

It remains to show that the range of f_π can be bounded in the size of the problem instance (and that this bound can be computed effectively). Then, Lemma 12 implies that the length of the paths $q^I_i(m_i, m'_i)q^F_i \in \mathcal{A}_i$ can be bounded as well, and by induction this holds for every path $q^I_j(m_j, m'_j)q^F_j \in \mathcal{A}_j$, $i \le j < k$. Thus, given an instance $(\mathcal{K}, \mathcal{A}_0, \ldots, \mathcal{A}_{k-1})$ of the path problem, a decision algorithm can first compute the bound on the length of the paths and then enumerate all paths of length restricted by the (computed) bound and check whether their labels satisfy the required conditions.

To show that the range of f_π is bounded, recall that the depth of m_i and m'_i is (effectively) bounded in the size of the problem instance, and thus so are l_j and l'_j for every $j \le r$. Now Proposition 6 implies that the range of f_π is bounded in the size of the problem instance. We conclude:

Proposition 7. PATHPROBLEM *is decidable.*

From this, Theorem 2 follows immediately.

We note that the runtime of the decision algorithm proposed here is nonelementary in the number of receive-send actions. This is because the index of the solvability preserving and path truncation ordering grows nonelementary in the number of receive-send actions. Thus, it remains to investigate whether the complexity lower bound shown in the following section is tight.

11 A complexity lower bound

We prove the following theorem.

Theorem 3. *For transducer-based protocols,* ATTACK *is PSPACE-hard.*

We first show:

Theorem 4. PATHPROBLEM *is a PSPACE-hard problem.*

This is done by reduction from the finite automata intersection problem, which is known to be PSPACE-complete [25].

Recall that a deterministic finite automaton $\mathcal{B}$ is a tuple $(Q, \Sigma, q^I, \delta, F)$, where Q is a finite set of states, Σ a finite alphabet, $q^I \in Q$ the initial state, $F \subseteq Q$ the set of final states, and δ a partial mapping taking a tuple $(q, a) \in Q \times \Sigma$ to a state $q' \in Q$. If δ is extended to words with $\delta(q, \varepsilon) := q$ and $\delta(q, aw) := \delta(\delta(q, a), w)$, then w is accepted by $\mathcal{B}$ if $\delta(q^I, w) \in F$.

The *finite automata intersection problem* is defined as follows: Given $k \ge 0$ deterministic finite automata $\mathcal{B}_0, \ldots, \mathcal{B}_{k-1}$ with $\mathcal{B}_i = (Q_i, \Sigma, q^I_i, \delta_i, F_i)$, decide whether there exists a word $w \in \Sigma^*$ accepted by $\mathcal{B}_i$ for all $i < k$.[6]

Given the $\mathcal{B}_i$'s, we define the corresponding path problem as follows. The set $\mathcal{N}$ of atomic messages is $\Sigma \cup \{a, \mathsf{secret}\}$, where a and secret are new atomic messages. The initial intruder knowledge $\mathcal{K}$ is Σ. The message transducers $\mathcal{A}_i$ will correspond to $\mathcal{B}_i$, and they work as follows: $\mathcal{A}_0$ is constructed in such a way that $\mathcal{A}_0$ outputs $\mathsf{enc}_a(w)$ on input w iff w is accepted by $\mathcal{B}_0$. That is, $\mathcal{A}_0$ works just like $\mathcal{B}_0$ except that it encrypts w with a. The message transducers $\mathcal{A}_i$, $0 < i < k-1$, do the same, but they expect the input messages to be of the form $\mathsf{enc}_a(w)$, and they only output this message if w is accepted by $\mathcal{B}_i$. Finally, $\mathcal{A}_{k-1}$ works in the same way but also returns secret if w is accepted by $\mathcal{B}_{k-1}$. The encryption of w guarantees that the intruder must send the same word w ($\mathsf{enc}_a(w)$) to all the $\mathcal{A}_i$'s. Without encryption, the intruder could send different words w' to every $\mathcal{A}_i$. Formally, $\mathcal{A}_0$ is

$$(Q_0 \cup \{q^0_0, q^1_0\}, \Sigma_{\mathcal{N}}, \{q^0_0\}, \Delta_0, \{q^1_0\})$$

with

$$\begin{aligned}\Delta_0 := {} & \{(q^0_0, \varepsilon, \mathsf{enc}_a(, q^I_0)\} \cup \\ & \{(q, b, b, q') \mid \delta_0(q, b) = q'\} \cup \\ & \{(q, \varepsilon,), q^1_0) \mid q \in F_0\}\end{aligned}$$

and for $0 < i < k-1$, $\mathcal{A}_i$ is

$$(Q_i \cup \{q^0_i, q^1_i\}, \Sigma_{\mathcal{N}}, \{q^0_i\}, \Delta_i, \{q^1_i\})$$

with

$$\begin{aligned}\Delta_i := {} & \{(q^0_i, \mathsf{enc}_a(, \mathsf{enc}_a(, q^I_i)\} \cup \\ & \{(q, b, b, q') \mid \delta_i(q, b) = q'\} \cup \\ & \{(q,),), q^1_i) \mid q \in F_i\}.\end{aligned}$$

Finally, $\mathcal{A}_{k-1}$ is defined just as $\mathcal{A}_i$ for $0 < i < k-2$ except that the transition $(q^1_i, \varepsilon, \mathsf{secret}, q^2_i)$ and the state q^2_i is

[6] If k is fixed, this problem can be decided in polynomial time.

added and q_i^2 is the only final state of $\mathcal{A}_{k-1}$. Now it is not hard to verify that there exists a word $w \in \Sigma^*$ accepted by $\mathcal{B}_i$ for all $i < k$ iff the instance $(\mathcal{K}, \mathcal{A}_0, \ldots, \mathcal{A}_{k-1})$ of the path problem has a solution. This proves Theorem 4.

We can basically employ the same argument for the problem ATTACK. We simply conjoin the message transducers $\mathcal{A}_i$ into one extended message transducer $\mathcal{A} = (Q, \Sigma_{\mathcal{N}}, \Delta, (I_0, \ldots, I_k))$, where Q and Δ are the union of the states and transitions of the $\mathcal{A}_i$'s, respectively, and $I_i := \{q_i^0\}$ for $i < k$, and $I_k := \{q_{k-1}^2\}$. We assume that the set of states of the $\mathcal{A}_i$'s are disjoint, except that we identify q_{i-1}^1 and q_i^0 for $0 < i < k$. Again, it is easy to see that there exists a word $w \in \Sigma^*$ accepted by $\mathcal{B}_i$ for all $i < k$ iff the transducer-based protocol with one principal defined by $\mathcal{A}$ allows a successful attack. Thus, Theorem 3 follows.

12 Conclusion

We have introduced a generic protocol model for analyzing the security of open-ended protocols, i.e., protocols with open-ended data structures, and investigated the decidability of different instances of this model. In one instance, receive-send actions are modeled by sets of rewrite rules. We have shown that in this instance, security is undecidable. This result indicated that to obtain decidability, principals should only have finite memory and should not be able to compare or copy messages of arbitrary size. This motivated our transducer-based model, which complies with these restrictions but still captures certain open-ended protocols. We have shown that in this model security is decidable and PSPACE-hard.

While in this paper we have concentrated on the shared key setting and secrecy properties, we conjecture that our results carry over rather easily to public key encryption and authentication.

An open problem is the establishment of a tight complexity bound. So far, the decision algorithm proposed is nonelementary in the number of receive-send actions and is merely based on the fact that in attacks the length of computations in receive-send actions and the size of messages can be bounded. Thus, it also remains to develop a more practical algorithm. As pointed out in Sect. 4.1, another promising future direction is to combine the transducer-based model with the models for closed-ended protocols and to devise tree transducers suitable for describing receive-send actions. We will also try to incorporate complex keys since they are used in many protocols. While we strongly conjecture that the proof techniques and the results presented here carry over to the case where keys are messages of bounded size, instead of only atomic messages, the techniques do not apply in the general case where no bound is put on the size of keys.

Acknowledgements. Thanks to Thomas Wilke and the anonymous referees for helpful comments on this work and to Catherine Meadows for directing me to the paper by Pereira and Quisquater.

References

1. Abadi M, Gordon AD (1997) A calculus for cryptographic protocols: the spi calculus. In: 4th ACM conference on computer and communications security. ACM Press, New York, pp 36–47
2. Amadio RM, Charatonik W (2002) On name generation and set-based analysis in the Dolev-Yao model. In: Brim L, Jancar P, Kretinsky M, Kucera A (eds) 13th international conference on concurrency theory (CONCUR 2002). Springer, Berlin Heidelberg New York, 2421:499–514
3. Amadio RM, Lugiez D, Vanackère V (2001) On the symbolic reduction of processes with cryptographic functions. Technical Report RR-4147, INRIA
4. Armando A, Basin D, Bouallagui M, Chevalier Y, Compagna L, Mödersheim S, Rusinowitch M, Turuani M, Viganò L, Vigneron L (2002) The AVISS security protocol analysis tool. In: Brinksma E, Larsen KG (eds) 14th international conference on computer aided verification (CAV 2002). Lecture notes in computer science, vol 2404. Springer, Berlin Heidelberg New York, pp 349–353
5. Ateniese G, Steiner M, Tsudik G (1998) Authenticated group key agreement and friends. In: Proceedings of the 5th ACM conference on computer and communication security (CCS'98), San Francisco. ACM Press, New York, pp 17–26
6. Barrett DJ, Silverman R, Loukides M (2001) SSH, the Secure Shell: the definitive guide. O'Reilly, UK
7. Bellare M, Rogaway P (1994) Entity authentication and key distribution. In: Stinson D (ed) Advances in Cryptology – Crypto '93, 13th annual international cryptology conference. Lecture notes in computer science, vol 773. Springer, Berlin Heidelberg New York
8. Boreale M (2001) Symbolic trace analysis of cryptographic protocols. In: Automata, languages and programming, 28th international colloquium (ICALP 2001). Lecture notes in computer science, vol 2076. Springer, Berlin Heidelberg New York, pp 667–681
9. Boreale M, Buscemi MG (2002) Experimenting with STA, a tool for automatic analysis of security protocols. In: Proceedings of the 2002 ACM symposium on applied computing (SAC 2002). ACM Press, New York, pp 281–285
10. Bryans J, Schneider SA (1997) CSP, PVS, and a recursive authentication protocol. In: DIMACS workshop on formal verification of security protocols
11. Bull JA, Otway DJ (1997) The authentication protocol. Technical Report DRA/CIS3/PROJ/CORBA/SC/1/ CSM/436-04/03, Defence Research Agency, Malvern, UK
12. Burrows M, Abadi M, Needham R (1990) A logic of authentication. ACM Trans Comput Syst 8(1):18–36
13. Cervesato I, Durgin NA, Lincoln PD, Mitchell JC, Scedrov A (1999) A meta-notation for protocol analysis. In: 12th IEEE Computer Security Foundations Workshop (CSFW-12)
14. Clark J, Jacob J (1997) A survey of authentication protocol literature. Web Draft Version 1.0 available at `http://citeseer.nj.nec.com/`
15. Cohen E (2000) TAPS: First-order verification of cryptographic protocols. In: 13th Computer Security Foundations Workshop (CSFW 2000). IEEE Press, New York, pp 144–158 (2002) Goubault-Larrecq J (ed) J Telecommun Inf Technol 4:5–15
16. Comon H, Shmatikov V (2002) Is it possible to decide whether a cryptographic protocol is secure or not? In: Goubault-Larrecq J (ed) J Telecommun Inf Technol (Special Issue on Cryptographic Protocol Verification) 4:5–15
17. Dolev D, Even S, Karp RM (1982) On the security of ping-pong protocols. Inf Control 55:57–68
18. Dolev D, Yao AC (1983) On the security of public-key protocols. IEEE Trans Inf Theory 29(2):198–208
19. Durante A, Focardi R, Gorrieri R (1999) Cvs: a compiler for the analysis of cryptographic protocols. In: Syverson P (ed) Proceedings of the 12th IEEE Computer Security Foundations Workshop (CSFW'99). IEEE Press, New York, pp 203–212
20. Durgin NA, Lincoln PD, Mitchell JC, Scedrov A (1999) Undecidability of bounded security protocols. In: Workshop on formal methods and security protocols (FMSP'99)

21. Even S, Goldreich O (1983) On the security of multi-party ping-pong protocols. In: IEEE symposium on foundations of computer science (FOCS'83), pp 34–39
22. Ferguson N, Schneier B (2000) A cryptographic evaluation of IPsec. Technical report. http://www.counterpane.com/ipsec.pdf
23. Fiore MP, Abadi M (2001) Computing symbolic models for verifying cryptographic protocols. In: 14th Computer Security Foundations Workshop (CSFW-14). IEEE Press, New York, pp 160–173
24. Freier A, Karlton P, Kocher P (1996) The SSL protocol version 3.0. draft-ietf-tls-ssl-version3-00.txt, 18 November 1996
25. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, San Francisco
26. Harkins D, Carrel D (1998) The Internet Key Exchange (IKE), The Internet Engineering Task Force, November 1998. RFC 2409
27. Huima A (1999) Efficient infinite-state analysis of security protocols. In: Workshop on formal methods and security protocols (FMSP'99)
28. Kohl J, Neuman B, Ts'o T (1994) The evolution of the Kerberos authentication service. In: Brazier F, Johansen D (eds) Distributed open systems. IEEE Press, New York
29. Lowe G (1995) An attack on the Needham-Schroeder public-key authentication protocol. Inf Process Lett 56:131–133
30. Lowe G (1996) Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In: Tools and algorithms for the construction and analysis of systems (TACAS 1996). Lecture notes in computer science, vol 1055. Springer, Berlin Heidelberg New York, pp 147–166
31. Meadows C (2000) Extending formal cryptographic protocol analysis techniques for group protocols and low-level cryptographic primitives. In: Degano P (ed) Proceedings of the 1st workshop on issues in the theory of security (WITS'00), pp 87–92
32. Meadows C (2000) Open issues in formal methods for cryptographic protocol analysis. In: Proceedings of DISCEX 2000. IEEE Press, New York, pp 237–250
33. Millen JK, Shmatikov V (2001) Constraint solving for bounded-process cryptographic protocol analysis. In: Proceedings of the 8th ACM conference on computer and communications security. ACM Press, New York, pp 166–175
34. Mitchell J, Mitchell M, Stern U (1997) Automated analysis of cryptographic protocols using Murphi. In: Proceedings of the 1997 IEEE symposium on security and privacy. IEEE Press, New York, pp 141–151
35. Otway D, Rees O (1987) Efficient and timely mutual authentication. Oper Syst Rev 21(1):8–10
36. Paulson LC (1997) Mechanized proofs for a recursive authentication protocol. In: 10th IEEE Computer Security Foundations Workshop (CSFW-10), pp 84–95
37. Paulson LC (1997) Proving properties of security protocols by induction. In: 10th Computer Security Foundations Workshop (CSFW-10). IEEE Press, New York, pp 70–83
38. Pereira O, Quisquater J-J (2001) A security analysis of the Cliques Protocols suites. In: Proceedings of the 14th IEEE Computer Security Foundations Workshop (CSFW-14), pp 73–81
39. Rusinowitch M, Turuani M (2001) Protocol insecurity with finite number of sessions is NP-complete. In 14th IEEE Computer Security Foundations Workshop (CSFW-14), pp 174–190
40. Schneider S (1996) Security Properties and CSP. In: Proceedings of the 1996 IEEE symposium on security and privacy. IEEE Press, New York, pp 174–187
41. Song DX, Berezin S, Perrig A (2001) Athena: a novel approach to efficient automatic security protocol analysis. J Comput Secur 9(1/2):47–74
42. Steiner M, Tsudik G, Waidner M (1998) CLIQUES: A new approach to key agreement. In: IEEE international conference on distributed computing systems. IEEE Press, New York, pp 380–387
43. Zhou J (1999) Fixing a security flaw in IKE protocols. Electron Lett 35(13):1072–1073

Appendices

A The recursive authentication protocol

The recursive authentication protocol (RA protocol) was proposed by Bull and Otway [11], and it extends the authentication protocol by Otway and Rees [35] in that it allows one to establish session keys between an a priori unbounded number of principals in one protocol run. Our description of the RA protocol follows Paulson [36].

In the RA protocol one assumes that a key distribution server S shares long-term keys with the principals. In Fig. 1 a typical protocol run is depicted. In this run, A wants to establish a session key with B and B wants to establish a session key with C. The number of principals involved in a protocol run is not bounded. In particular, C could send a message to some principal D in order to establish a session key with D and D could continue and send a message to E and so on. In the protocol run depicted in Fig. 1, we assume that C does not want to talk to another principal and therefore sends a message to the key distribution server S who is involved in every protocol run.

In Fig. 1, K_a denotes the long-term keys shared between A and S. Similarly, K_b and K_c are the long-term keys shared between B, C, and S, respectively. With N_a, N_b, and N_c we denote nonces (i.e., random numbers) generated by A, B, and C, respectively. Finally, K_{ab}, K_{bc}, and K_{cs} are the session keys generated by the server and

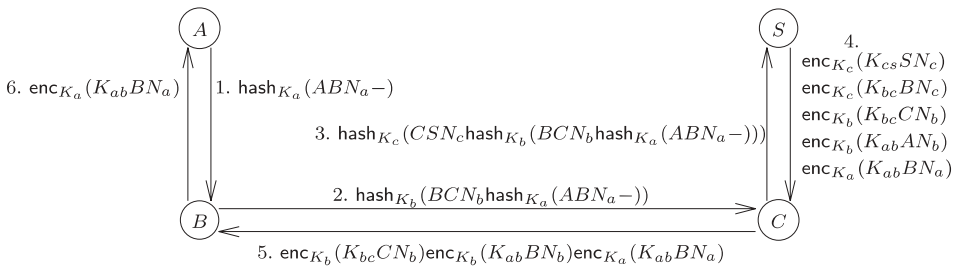

Fig. 1. The recursive authentication protocol

used by the principals for secure communication between A and B, B and C, and C and S, respectively. The numbers (1–6) attached to the messages only indicate the order in which the messages are sent and do not belong to the protocol.

The message $\mathsf{hash}_k(m)$ for a key k and some message m stands for $\mathsf{hash}(km)m$. In other words, $\mathsf{hash}_k(m)$ contains a keyed hash of m (the message authentication code of m) plus m itself.

We now take a closer look at the messages exchanged between the principals in the order they are sent: In the first messages (1), principal A indicates that she requests a session key from the server for secure communication with B. The symbol "$-$" says that this message started the protocol run. Now, in the second message (2), B sends something similar to C but with A's message instead of "$-$", indicating that he wants to share a session key with C. As mentioned, this step could be repeated as many times as desired, yielding an ever-growing stack of requests. The process is terminated if one principal contacts S. In this example, we assume that C does not request another session key and therefore sends the message received from B to S (3). This message is now processed by S.

The outer two hashes indicate that C has called S and was called by B. Therefore, the server generates fresh keys K_{cs} and K_{bc}, intended to be used as a session key between C and S and between B and C, respectively. Then, S prepares certificates $\mathsf{enc}_{K_c}(K_{cs}SN_c)$ and $\mathsf{enc}_{K_c}(K_{bc}BN_c)$, which together with the certificates prepared later will be sent to C. Note that the key K_{cs} is redundant since C and S already share a key. But including it allows one to treat the last principal in the chain like all others, except the first, who only receives one session key. Having dealt with C's request, the server discards the outer level and proceeds with the message $\mathsf{hash}_{K_b}(BCN_b\ \mathsf{hash}_{K_a}(ABN_a-))$. It says that B has called C and was called by A. Therefore, the server prepares two certificates $\mathsf{enc}_{K_b}(K_{bc}CN_b)$ and $\mathsf{enc}_{K_b}(K_{ab}AN_b)$, both intended for B, the former containing the session key for communication with C and the latter containing the one for communication with A. Note that K_{bc} is the same key sent to C, and K_{ab} is a fresh key generated by the server. It remains to process $\mathsf{hash}_{K_a}(ABN_a-)$. It indicates that A requests a session key for communication with B, and because it contains "$-$", A must have initiated the protocol run. Thus, the server prepares only one certificate: $\mathsf{enc}_{K_a}(K_{ab}BN_a)$, where K_{ab} is the same key as the one sent to B. Having prepared all necessary certificates, the server sends all of them to C (4). The line break is only for layout purposes and does not have any meaning in the protocol.

Principal C accepts the first two certificates, extracts the two session keys, and forwards the rest of the message to its predecessor in the chain (5). Then, B does the same, and forwards the last certificate to A (6).

B Modeling the recursive authentication protocol

In this section, a formalization of the key distribution server in terms of the transducer-based protocol model is provided. Although in the recursive authentication protocol the number of receive-send actions performed in one protocol run is unbounded, in our model we assume a fixed bound – extending the transducer-based protocol model to handle an unbounded number of receive-send actions would lead to undecidability, just as for an unbounded number of sessions. Nevertheless, even with such a fixed bound it is still necessary to model open-ended data structures. The most important reason is that the intruder can generate messages with an unbounded number of data fields, which must be processed by the principals. Also note that in other protocols, such as IKE, open-ended data structures may occur even without an intruder.

To provide a formal description of the RA protocol in the transducer-based model, we simplify the messages exchanged between the principals a bit. We assume that instead of $\mathsf{hash}(km)m$ a message of the form $\mathsf{hash}(km)$ is sent. With this, we implicitly assume that, together with a hash, the corresponding message is sent. To make sure that the intruder gets hold of m, even if only $\mathsf{hash}(km)$ is sent, the principals will send some additional information. For instance, in the first message, A would send $ABN_a\mathsf{hash}(K_aABN_a-)$. In the second step, B expects a message of the form $\mathsf{hash}(K_aABN_a-)$, i.e., from the messages A sent the first part, namely, ABN_a, is removed by the intruder. Principal B does not need this information because he can extract it from $\mathsf{hash}(K_aABN_a-)$. The message sent out by B is of the form $BCN_b\mathsf{hash}(K_bBCNB\ \mathsf{hash}(K_aABN_a-))$.

In what follows, we give a formal description of the key distribution server S within the transducer-based model. The server is the only principal who needs to process open-ended data structures. In their first receive-send action, the other principals take only the message received, examine the outermost request (ignoring the others), and add their request, or, in the second receive-send action, only extract their certificates from the message received and forward the remaining message, regardless of the content. For this reason, we just present a description of the server in order to illustrate the use of the transducer-based protocol model.

Let $P_0, \ldots, P_n$ be the principals participating in the RA protocol. We assume that $P_n = S$ is the server. Every P_i, $i < n$, shares a long-term key K_i with S. The nonce sent by P_i in the request message is denoted N_i, $i < n$.

The set of atomic messages is $\mathcal{N} := \{P_i \mid i \leq n\} \cup \{K_i \mid i < n\} \cup \{N_i \mid i < n\} \cup \{K_{jj'} \mid j < n,\ j' \leq n\} \cup \{-\}$. The initial intruder knowledge $\mathcal{K}$ contains all the principal names plus the symbol "$-$" and the empty word ε. One could also add keys K_i and nonces N_i in case the intruder controls P_i.

1. S reads the first request and generates the corresponding certificate. For every $a \in \mathcal{N}$ and $i < n$,
$$(\mathsf{start}, \bot, \bot) \xrightarrow[\mathsf{enc}_{K_i}(K_{in}Sa)]{\mathsf{hash}(K_iP_iSa} (\mathsf{read}, P_i, a);$$
2. P_i has initiated the protocol run, and therefore, no additional certificate needs to be generated: For every $i < n$ and $a \in \mathcal{N}$,
$$(\mathsf{read}, P_j, a) \xrightarrow[\varepsilon]{-} (\mathsf{readpar}, \bot, \bot);$$
3. The remaining certificate for P_i is generated and a new one for $P_{i'}$: For every $i, i' < n$ and $a_0, a_1 \in \mathcal{N}$,
$$(\mathsf{read}, P_i, a_0) \xrightarrow[\mathsf{enc}_{K_i}(K_{i'i}P_{i'}a_0)\mathsf{enc}_{K_{i'}}(K_{i'i}P_ia_1)]{\mathsf{hash}(K_{i'}P_{i'}P_ia_1} (\mathsf{read}, P_{i'}, a_1);$$
4. The remaining closing parentheses are read:
$$(\mathsf{readpar}, \bot, \bot) \xrightarrow[\varepsilon]{)} (\mathsf{readpar}, \bot, \bot);$$
5. S is done if the last closing parenthesis is read:
$$(\mathsf{readpar}, \bot, \bot) \xrightarrow[\varepsilon]{)} (\mathsf{accept}, \bot, \bot).$$

Fig. 2. Transitions of the key distribution server $\mathcal{A}_n$

The extended message transducer $\mathcal{A}_n$ for the server S is defined as follows. Recall that S expects messages of the form

$$\mathsf{hash}(a_iP_iSb_i\mathsf{hash}(a_jP_jP_ib_j\mathsf{hash}(\cdots)\cdots)$$

with $a_i, b_i, a_j, b_j, \ldots \in \mathcal{N}$. The states of S consist of three components. The first takes the values start, read, $\mathsf{readpar}$, and accept. In state start, $\mathcal{A}_n$ reads the first symbols of the message, checks whether this message is really addressed to S, and generates the first certificates. In state read, $\mathcal{A}_n$ processes the rest of the requests. At the end, $\mathcal{A}_n$ needs to read remaining closing parentheses. This is done in state $\mathsf{readpar}$. If everything is ok, S goes into the state accept. In the second component, $\mathcal{A}_n$ memorizes whose certificates are to be generated, and the third component stores the corresponding nonce.

The transitions of S are depicted in Fig. 2. To increase readability, a transition (p, v, w, q) is written in the following form:

$$p \xrightarrow[w]{v} q\,.$$

Note that words read/written in one transition are not necessarily messages, i.e., the number of parentheses may be unbalanced.

Since S only performs a single action, we only need to define two sets I_0 and I_1: $I_0 := \{(\mathsf{start}, \bot, \bot)\}$, $I_1 := \{(\mathsf{accept}, \bot, \bot)\}$. It is easy to turn $\mathcal{A}_n$ into a transducer with letter transitions such that the conditions on message transducers are satisfied.

Note that the nonces sent by the principals to S must be stored by S since they may occur in two certificates. That is, S must copy submessages. As discussed in Sect. 4.1, because transducers have only finite memory, this is only possible if the submessages to be copied have bounded size. Therefore, S, as modeled here, assumes nonces to be atomic.