INSTITUT FÜR INFORMATIK

UND PRAKTISCHE MATHEMATIK

On the Decidability of Cryptographic Group Protocols

Ralf Küsters

Bericht Nr. 0201
February 2002

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

KIEL

Institut für Informatik und Praktische Mathematik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

On the Decidability of Cryptographic Group Protocols

Ralf Küsters

Bericht Nr. 0201
February 2002

e-mail: kuesters@ti.informatik.uni-kiel.de

On the Decidability of Cryptographic Group Protocols

Ralf Küsters
Institut für Informatik und Praktische Mathematik
CAU Kiel, Germany
kuesters@ti.informatik.uni-kiel.de

Abstract

We propose a protocol model in which principals are described by transducers (Mealy machines), i.e., finite automata with output. This allows the principals to process messages with an unbounded number of data fields, a requirement typical for group protocols. However, as in other models, a finite number of sessions and protocol steps is assumed. For this setting, we show security to be decidable provided that the nesting depth of encryptions and hashes performed by the intruder is bounded.

1 Introduction

Formal methods have turned out to be very successful in analyzing the security of cryptographic protocols. Using these methods, many flaws have been found in published protocols. By now a large variety of different methods and tools for cryptographic protocol analysis have been devised (see [13] for an overview), and researchers have investigated the decidability of security for different classes of protocols and intruders. The decidability results can be summarized as follows. In general, if one allows for an unbounded number of sessions, security is undecidable [8, 7, 1]. One exception is when the message size is bounded and nonces are disallowed [7] (then security is EXPTIME-complete); another exception is when pairing is disallowed [6, 1] (then security is in P). If the number of sessions is bounded and nonces cannot be generated, security becomes decidable, even if pairing is allowed and the message size is unbounded, and NP-hard [17, 1, 10]; Rusinowitch and Turuani [17] show that in this setting the problem is in NP, even with complex keys.

Most of the mentioned work concentrates on a class of protocols with a fixed number of principals (e.g., initiator and responding in a key exchange protocol), a fixed number of steps, and a fixed and finite format of the messages exchanged between the principals. However, group protocols, such as the recursive authentication protocol [4] and the A-GDH.2 protocol [2] (which is part of the CLIQUES project [18]), do not fall into this class of protocols. In fact, two main features of group protocols are that

1. the number of principals and steps is not fixed, and
2. principals need to process messages with an unbounded number of data fields.

The latter feature also occurs in protocols such as the SET Protocol and the IKE Protocol, and as pointed out in [13], this kind of "open-endedness" is security-relevant: For example, Zhou [19] and independently Ferguson and Schneier [9] found an attack on the IKE Protocol in which an intruder could trick an initiator into agreeing on the wrong security association by making use of the fact that only part of the security association is actually used in IKE.

As an example of a group protocol we take a closer look at the recursive authentication protocol by Bull and Otway [4], which extends the authentication protocol by Otway and Rees [14] in that it allows to establish session keys between an (a priori) unbounded number of principals in one protocol run. The protocol works as follows: We assume that the authentication server S shares long-term keys with the principals. If A wants to establish a session key with B, she sends a request message to B. In the Otway-Rees protocol, B would now contact the server S, which in turn would generate a session key and distribute it to A and B. In the recursive authentication protocol, B can also contact another principal, say C, and so on, until a principal contacts S. During each round a principal adds his name and a fresh nonce to the ever-growing request message. When the server obtains this message, it generates fresh keys for the pairs of principals requesting session keys, i.e., S generates a key for A and B, one for B and C, and so forth. Finally, S distributes the keys to the principals, encrypted with the respective long-term keys. Thus, the number of steps performed and principals involved in one protocol run is unbounded. Also, the server needs to process the request message, which has an unbounded number of data fields, namely an unbounded sequence of pairs of principals requesting session keys.

As pointed out by Meadows [13], there has only be very little work on applying formal methods to cryptographic group protocols (or more generally, open-ended protocols). To the best of our knowledge, the only contributions are the following ones: The recursive authentication protocol has been analyzed by Paulson [15], using the Isabelle theorem prover, as well as by Bryans and Schneider [3], using the PVS theorem prover; the A-GDH.2 protocol has been analyzed by Meadows [12] with the NRL Analyzer, and manually by Pereira and Quisquater [16], based on a model similar to the strand spaces model.

In this paper, we try to devise a model for cryptographic group protocols in which security is decidable, and which still captures a large class of protocols. Our model extends the ones in [17, 1, 10] in that actions performed by principals are modeled by transducers (Mealy machines), i.e., finite automata with output. This enables principals to process messages with an unbounded number of data fields. As mentioned, this is an important feature of group protocols (see 2. above), which the models usually considered in the literature cannot capture; for instance, the models by Rusinowitch and Turuani [17], and Amadio et al. [1], where actions are described by single rewrite rules and processes without loops, respectively. However, as in [17, 1], we assume a finite number of sessions, and we also assume a finite number of steps within one session. Otherwise, principals could perform an unbounded number of steps, which, as mentioned at the beginning, usually leads to undecidability unless one imposes strong restrictions on the intruder and the messages. In other words, for the sake of decidability one of the features of group protocols, namely the unbounded number of

principals and steps in one protocol run (see 1. above), is not captured.

We show that in our model, security is decidable provided that the number of nested encryptions and hashes performed by the intruder is bounded. Note that by concatenating messages (pairing), the intruder can still derive an infinite number of messages from his knowledge and can still perform an infinite number of encryptions and hashes, only the nesting depth is restricted.

The structure of the paper is as follows. Section 2 provides a description of the recursive authentication protocol, which later on (Section 6) will be formalized in our transducer-based model. In Section 3, we define a generic model for describing group protocols. The (generic) actions performed by principals in one step of a protocol are modeled by binary relations over the message space, i.e., principals are allowed to perform any (nondeterministic) computation within one step. However, for reasons discussed above, we only allow a finite number of sessions, and in every session only a finite number of steps. In Section 4, we consider certain instances of the generic model. In these instances, actions are modeled by sets of rewrite rules, which can be applied nondeterministically to the input message. Security in these instances is shown to be undecidable. In Section 5, we introduce the transducer-based model, in which actions are described by transducers. Then, in Section 6, we provide a formal description of the recursive authentication protocol in the transducer-based model. Section 7 contains the actual decidability result. Finally, we conclude in Section 8.

2 The Recursive Authentication Protocol

Following Paulson [15], we now provide a more precise description of the recursive authentication protocol. We first introduce some notation and then present the protocol run already mentioned in the introduction in more detail.

Let $\mathtt{hash}(m)$ be the hash of m, and $\mathtt{hash}_k(m)$ the message $\mathtt{hash}(km)m$, where km is the message obtained from k and m by concatenation, and $\mathtt{hash}(km)m$ is the concatenation of $\mathtt{hash}(km)$ and m. In the protocol, k will be a long-term key shared between the server S and a principal. It is used by the server to identify the principals. In other words, $\mathtt{hash}_k(m)$ contains the message m plus the message authentication code for m computed using k.

Figure 1 depicts a typical protocol run. First, A contacts B, then B contacts C, and C contacts the server S. Then, the session keys generated by the server are distributed among the principals by sending messages to the principals in reverse order, i.e., S first sends a message (containing all the keys) to C, C extracts his key, and sends the remaining message to B, who does the same, and sends the remaining message to A. More precisely, the messages exchanged are of the following form.

Principal A first sends a message to B:

1. $A \to B : \mathtt{hash}_{K_a}(ABN_a-)$,

where K_a is a long-term key shared between A and S, N_a is a fresh nonce generated by A, and "$-$" indicates that this message started the protocol run. In this message A indicates that she requests a session key from the server for secure communication

with B. Now, B sends something similar to C but with A's message instead of "$-$", indicating that he wants to share a session key with C.

2. $B \to C : \mathtt{hash}_{K_b}(BCN_b\mathtt{hash}_{K_a}(ABN_a-))$.

This step can 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. $C \to S : \mathtt{hash}_{K_c}(CSN_c\mathtt{hash}_{K_b}(BCN_b\mathtt{hash}_{K_a}(ABN_a-)))$.

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 $\mathtt{enc}_{K_c}(K_{cs}SN_c)$ and $\mathtt{enc}_{K_c}(K_{bc}BN_c)$, 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 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 $\mathtt{hash}_{K_b}(BCN_b\mathtt{hash}_{K_a}(ABN_a-))$. It says that B has called C and was called by A, so the server prepares two certificates $\mathtt{enc}_{K_b}(K_{bc}CN_b)$ and $\mathtt{enc}_{K_b}(K_{ab}AN_b)$. 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 the message $\mathtt{hash}_{K_a}(ABN_a-)$. 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: $\mathtt{enc}_{K_a}(K_{ab}BN_a)$, 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. (The line break in the following message is only for layout purposes and does not have any meaning in the protocol.)

4. $S \to C$:$\mathtt{enc}_{K_c}(K_{cs}SN_c)\mathtt{enc}_{K_c}(K_{bc}BN_c)\mathtt{enc}_{K_b}(K_{bc}CN_b)\mathtt{enc}_{K_b}(K_{ab}AN_b)$
$\mathtt{enc}_{K_a}(K_{ab}BN_a)$

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. Then, B does the same, and forwards the last certificate to A:

5. $C \to B$:$\mathtt{enc}_{K_b}(K_{bc}CN_b)\mathtt{enc}_{K_b}(K_{ab}AN_b)\mathtt{enc}_{K_a}(K_{ab}BN_a)$

6. $B \to A$:$\mathtt{enc}_{K_a}(K_{ab}BN_a)$

Note that in steps 4 to 6, a principal just extracts his part of the message and simply forwards the rest. For the analysis of the protocol, one can therefore assume that every principal only reads his own certificates – the intruder can do the forwarding instead. This is how the protocol will be modeled in Section 6.

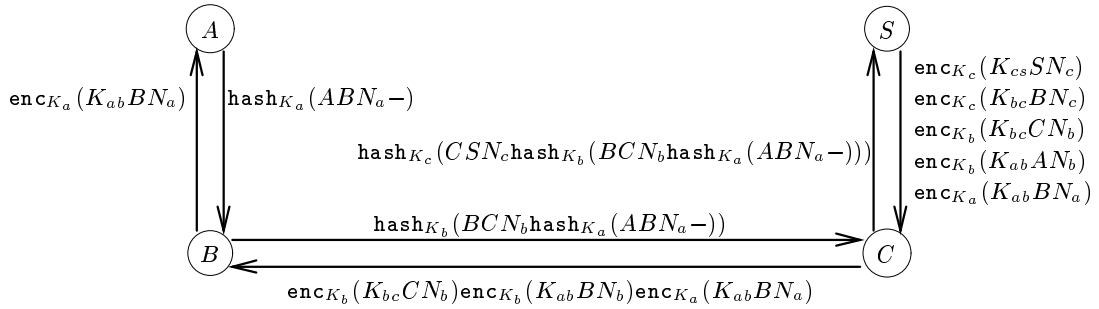

Figure 1: The Recursive Authentication Protocol

3 A Generic Protocol Model

Our main goal is to devise a formalism rich enough to describe a large class of cryptographic group protocols, but for which the existence of attacks is still decidable. Recall from the introduction that group protocols have two main features:

1. The number of principals and steps is not fixed, and
2. principals need to process messages with an unbounded number of data fields.

From a computational point of view, the first feature is very similar to a setting, where one allows an unbounded number of interleaved sessions, since in both cases the principals can perform an unbounded number of steps. However, the undecidablility results [7, 1] show that in such a model one cannot expect security to be decidable unless one imposes strong restrictions on the intruder and the messages. Therefore, in our generic model, we will only allow a finite number of sessions and within one session only a finite number of principals and steps.

As pointed out in the introduction, the second feature alone, i.e., principals processing an unbounded number of data fields, is already security-relevant (not only in group protocols), and therefore important to model. In our generic protocol model, actions performed by principals are therefore modeled as binary relations on the set of messages. The first component of such a relation is the input accepted by a principal, and the second component is the output. We use binary relations instead of partial functions to allow actions to be nondeterministic. The main question is what are suitable computational models for describing such relations. In the present work, we will consider sets of rewrite rules as well as transducers for this purpose, which, unlike single rewrite rules [17] or processes without loops [1], allow a principal to examine an unbounded number of data fields. However, only the transducers will yield the desired decidability result (provided that the nesting depth of encryptions and hashes performed by the intruder is bounded).

Let us now give a precise definition of our formalism.

3.1 Messages

The formal definition of messages is rather standard. 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 $m \cdot m' \in \mathcal{M}$;
- if $m \in \mathcal{M}$ and $a \in \mathcal{N}$, then $\mathtt{enc}_a(m) \in \mathcal{M}$;
- if $m \in \mathcal{M}$, then $\mathtt{hash}(m) \in \mathcal{M}$.

Usually, we simply write mm' for the composition $m \cdot m'$ of m and m'. Also, "$\cdot$" is considered an associative constructor, i.e., $m(m'm'') = (mm')m''$ for all messages $m, m', m'' \in \mathcal{M}$. Therefore, we can omit the parentheses. Finally, note that we only allow for atomic keys, i.e., in a message $\mathtt{enc}_a(\cdot)$, a is always an atomic message.

Let ε denote the *empty message* and $\mathcal{M}_\varepsilon := \mathcal{M} \cup \{\varepsilon\}$ the set of messages containing ε. Note that ε is not allowed inside encryptions or hashes, for instance, $\mathtt{enc}_a() \notin \mathcal{M}_\varepsilon$.

Later, we will also consider terms, i.e., messages with variables. Let $V := \{v_0, \ldots, v_{n-1}\}$ be a set of variables. Then a *term* t (over V) (also written $t(v_0, \ldots, v_{n-1})$) is a message over the atomic messages $\mathcal{N} \cup V$, where variables are not allowed as keys, i.e., terms of the form $\mathtt{enc}_v(\cdot)$ for some variable v are forbidden. Let $\mathcal{T}(V)$ denote the set of terms over V and $\mathcal{T}_\varepsilon(V) := \mathcal{T}(V) \cup \{\varepsilon\}$. For $t_0, \ldots, t_{n-1}, t \in \mathcal{T}_\varepsilon(V)$, $t[v_0/t_0, \ldots, v_{n-1}/t_{n-1}]$ denotes the term obtained from t by simultaneously substituting the variables v_i by t_i. A term $t' \in \mathcal{T}_\varepsilon(V)$ is a *subterm* of $t \in \mathcal{T}_\varepsilon(V)$ if there exists a term $t'' \in \mathcal{T}(V \cup \{v\})$, where v is a new variable occurring exactly once in t'', such that $t = t''[v/t']$. 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 encryptions and hashes in t, i.e.,

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

Given a subset of messages $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$ (the intruder's knowledge), the set of messages $\mathsf{d}(\mathcal{K})$ the intruder can derive from $\mathcal{K}$ 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 $\mathtt{enc}_a(m) \in \mathsf{d}(\mathcal{K})$ 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 $\mathtt{enc}_a(m) \in \mathsf{d}(\mathcal{K})$ (encryption);

- if $m \in \mathsf{d}(\mathcal{K})$ and $m \neq \varepsilon$, then $\mathtt{hash}(m) \in \mathsf{d}(\mathcal{K})$ (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 is well-known 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 this means (see, e.g., [1] for a proof):

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

In Section 5, in order to obtain our decidability result, we will restrict the intruders ability to derive new messages from its knowledge, by bounding the nesting depths of encryptions and hashes. Let h be a non-negative integer. For $\mathcal{K} \subseteq \mathcal{M}$, let $\mathsf{comp}(\mathcal{K})$ be the closure of $\mathcal{K}$ under composition of messages. Then the set of messages obtained from $\mathcal{K}$ by an arbitrary number of compositions and at most one nested encryption or hash is

$$\begin{aligned}\mathsf{csc}(\mathcal{K}) := \mathsf{comp}(&\mathcal{K} \cup \\ &\{\mathtt{enc}_a(m) \mid a \in \mathcal{K} \cap \mathcal{N} \text{ and } m \in \mathsf{comp}(\mathcal{K})\} \cup \\ &\{\mathtt{hash}(m) \mid m \in \mathsf{comp}(\mathcal{K})\}).\end{aligned}$$

The set of messages obtained by an arbitrary number of compositions and at most h nested encryptions and hashes is

$$\mathsf{syn}_h(\mathcal{K}) := \bigcup_{i \leq h} \mathsf{csc}^i(\mathcal{K})$$

with $\mathsf{csc}^0(\mathcal{K}) := \mathcal{K}$ and $\mathsf{csc}^{i+1}(\mathcal{K}) := \mathsf{csc}(\mathsf{csc}^i(\mathcal{K}))$. The set of messages that can be derived from $\mathcal{K}$ with at most h nested encryptions and hashes is

$$\mathsf{d}_h(\mathcal{K}) := \mathsf{syn}_h(\mathsf{an}(\mathcal{K})).$$

Note that the intruder can still perform an arbitrary number of encryptions and hashes. The only difference to $\mathsf{d}(\mathcal{K})$ is that the number of nested encryptions and hashes is bounded. We obtain the following bound on

$$\mathsf{depth}_h(\mathcal{K}) := \mathsf{max}\{\mathsf{depth}(m) \mid m \in \mathsf{d}_h(\mathcal{K})\}.$$

Observation 2 *Let $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$. Then, $\mathsf{depth}_h(\mathcal{K}) \leq \mathsf{max}\{\mathsf{depth}(m) \mid m \in \mathcal{K}\} + h$.*

3.2 Protocols

Protocols are described by sets of principals and every principal is defined by a sequence of actions. In a protocol run, the actions of a principal are applied one after the other: A principal first waits for input, performs his first action, and sends the resulting output. Then, he waits for the second input, performs the next action, and so forth. Since we are only interested in attacks (rather than "ordinary" protocol runs without an intruder), the definition of a protocol also contains the initial intruder knowledge. Formally, principals and protocols are defined as follows.

Definition 3 *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 steps to be performed by* Π*;*
- α *is a mapping assigning to every* $j \in \{0, \ldots, n-1\}$*, an action* $\alpha(j) \subseteq Q \times \mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon \times Q$*.*

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

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

In an attack on a protocol P, the 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 4 *Let* $P = (n, \{\Pi_i\}_{i<n}, \mathcal{K})$ *be a generic protocol with* $\Pi_i = (Q_i, I_i, n_i, \alpha_i)$*, for* $i < n$*. An* attack *on* P *is a tuple consisting of the following components:*

- *a set* $I \subseteq \{0, \ldots, n-1\}$ *(the (indices of) principals participating in the attack);*
- *a total ordering* $<$ *on the set* $\{(i,j) \mid i \in I, j < n_i\}$ *such that* $(i,j) < (i,j')$ *implies* $j < j'$ *(the execution order of the actions);*[1]
- *a mapping* ψ *assigning to every* (i,j)*,* $i \in I$*,* $j < n_i$*, a tuple*

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

with

  - $q_i^j, q_i^{j+1} \in Q_i$ *(the state of* i *before/after step* j*); and*
  - $m_i^j, {m'}_i^j \in \mathcal{M}_\varepsilon$ *(the input and output message in step* j*);*

such that

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

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

An h-attack*,* $h \geq 0$*, is defined just like an attack, but where* $\mathsf{d}(\cdot)$ *is replaced by* $\mathsf{d}_h(\cdot)$*, i.e., the intruder is only allowed to perform at most* h *nested encryptions and hashes. The same applies to the definition of* successful h*-attacks.*

[1] Although, we assume a linear ordering on the actions performed by a principal, we could as well allow partial orderings (as in [17]) without any impact on the decidability results.

The decision problems we are interested in are the following:

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

h-ATTACK: Given a protocol P, decide whether there exists a successful h-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 the problems ATTACK and h-ATTACK are decidable or not heavily depends on what kinds of actions a principal is allowed to perform. In the subsequent sections, we look at different instances of generic protocols, i.e., different computational models for actions, and study the problems ATTACK and h-ATTACK for the corresponding classes of protocols.

4 Undecidability Results

As a special instance of the multiset-rewriting setting [5], Rusinowitch and Turuani [17] define actions by single rewrite rules of the form $t \to t'$, where t and t' are terms.[2] A principal is a partially ordered finite set of such rules. A state of a principal is implicitly given by the values assigned to the variables occurring in the rewrite rules – different rules may share variables. A message m is transformed by an action of the form $t \to t'$ into the message $\sigma(t')$, where σ is a substitution with $m = \sigma(t)$. In [17], it is shown that in this setting ATTACK is an NP-complete problem.

In order to model principals, who can process messages with an infinite number of data fields, we extend the model by Rusinowitch and Turuani, and allow a principal to apply a finite (but a priori unbounded) sequence of rewrite rules to the message received. To this purpose, an action will be defined by a set of input, output, and so-called process rules. In this general setting, we only consider stateless principals. We have also considered a different model (see below), in which terms are required to be linear, i.e., every variable occurs at most once a term. In this case, we allow a principal to have an internal state.

A message is processed by an action as follows: First one of the input rules is applied, resulting in a new message. Then, non-deterministically, process rules are applied to this message, and finally, one of the output rules is used to produce the actual output.

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 and O are finite sets of rewrite rules (the input and output rules, respectively), and R is a finite set of rewrite rules (the process rules). 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 follow binary relation R_A on $\mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon$: $(m, m') \in$

[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 Section 3.1.

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 $i \leq n$, 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 actions can be described by rule-based actions is called *rule-based protocol*. Note that since in this setting principals do not have an internal state, 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 5 *For rule-based protocols and every $h \geq 0$, the problems* ATTACK *and* h-ATTACK *are undecidable.*

In fact, the proof shows that one principal, performing one action, suffices to obtain undecidability.

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

Given such a problem, we define the corresponding rule-based protocol P as follows: P has one principal performing one action $A = (I, O, R)$ with

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

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 Post's Correspondence Problem has a solution. Obviously, the intruder does not need to encrypt or hash messages. Thus, the reduction also works for h-attacks, $h \geq 0$.

However, the reduction does not work if we only allowed for linear terms, since $x = x$ is not a linear term. Nevertheless, if principals have unbounded memory to store one term (which then presents the internal state of a principal), even with linear terms we easily get undecidability. We first define this so-called linear-term protocols formally and then show undecidability.

A *linear-term action* A is a tuple (I, O, R) such that

- I is a set of rules of the form $s \to (t, t')$, where s, t, t' are linear terms;
- O is a set of rules of the form $(s, s') \to t$, where s, s', t are linear terms;
- R is a set of rules of the form $(s, s') \to (t, t')$, where s, s', t, t' are linear terms and for all substitutions σ, $|\sigma(t)| + |\sigma(t')| < |\sigma(s)| + |\sigma(s')|$.

Note that the state of a principal only depends on the current input and is independent of previous actions. Similar to rule-based actions, a linear-term action $A = (I, O, R)$ induces a binary relation $R_A \subseteq \mathcal{M}_\varepsilon \times \mathcal{M}_\varepsilon$ as follows: $(m, m') \in R_A$ iff there exist substitutions $\sigma_0, \sigma_1, \ldots, \sigma_n$ and rules $r_0, \ldots, r_n$ such that

- $r_0 \in I$, $r_n \in O$, and $r_i \in R$ for every $0 < i < n$;
- $\sigma_0(s_0) = m$ if $r_0 = s_0 \to (t_0, t_0')$;
- $\sigma_n(t_n) = m'$ if $r_n = (s_n, s_n') \to t_n$; and
- $\sigma_i(t_i) = \sigma_{i+1}(s_{i+1})$ and $\sigma_i(t_i') = \sigma_{i+1}(s_{i+1}')$ if $r_i = (s_i, s_i') \to (t_i, t_i')$, for every $0 < i < n$, and $r_0 = s_0 \to (t_0, t_0')$.

A generic protocol for which the actions are define by linear-term actions is called *linear-term protocol*.

Theorem 6 *For linear-term protocols and every $h \geq 0$, the problems* Attack *and* h-Attack *are undecidable.*

The proof is very similar to the one for rule-based protocols, and again is by a reduction from Post's Correspondence Problem. However, since we cannot test in one step whether the identity produced by the intruder in fact holds, this is done in two steps. First, one side of the identity is stored (input rule) and then in the next step, the stored term is compared with the other side of the identity (see rule $(x = y, \#y\#) \to (x = y, \#)$ below). Formally, given an instance of Post's Correspondence Problem as above, we define the corresponding linear-term protocol P as follows: P has one principal, who performs one action $A = (I, O, R)$ with

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

where x and y variables, and $\# \notin \Sigma$ is a new letter. The initial intruder knowledge is $\mathcal{K} := \Sigma \cup \{=\}$. It is easy to see that P allows a successful attack iff the instance of Post's Correspondence Problem has a solution. Again, the intruder does not need to encrypt or hash messages. Thus, the reduction also works for h-attacks, $h \geq 0$.

In the following section, we therefore restrict the principals to have only finite memory. More precisely, we use transducers to model actions. For this class of protocols we will show that h-Attack is decidable for every $h \geq 0$.

5 The Transducer-Based Protocol Model

We define an instance of the generic model, in which actions of principals are modeled by finite transducers.

Transducers (Mealy machines) are finite automata with output. Formally, they are defined as follows. 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 7 *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 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 $q_0(v_0, w_0)q_1(v_1, w_1)q_2 \dots (v_{n-1}, w_{n-1})q_n$ 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 non-empty 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 exists $p \in I$ and $q \in F$ with $(p, v, w, q) \in \mathcal{A}(I, F)\}$.

If $\Delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Omega \cup \{\varepsilon\}) \times Q$, $\mathcal{A}$ 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.

Lemma 8 *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* $\mathcal{A}'(S, T) = \mathcal{A}(S, T)$ *for every* $S, T \subseteq Q$*.*

Proof. Every transition of $\mathcal{A}$ is turned into a set of transitions such that first the input label is read letter by letter and then the output label is written letter by letter. This requires to introduce intermediate states. Formally, let $(p, v, w, q) \in \Delta$ with $v = v_0 \cdots v_{l-1}$, $v_i \in \Sigma$ for all $i < l$, and $w = w_0 \cdots w_{r-1}$, $w_i \in \Omega$ for all $i < r$; $l = 0$ or $r = 0$ is allowed. Assume $l > 1$ or $r > 1$ (otherwise the transition has the correct format). This transition can be replaced by the transitions $(p_i, v_i, \varepsilon, p_{i+1})$, $i < l$, and $(q_i, \varepsilon, w_i, q_{i+1})$, $i < r$, where $p_0 = p$, $q_r = q$, $p_l = q_0$, and the states $p_1, \dots, p_l, q_0, \dots, q_{r-1}$ are new. Replacing every transition in this way yields a transducer with letter transitions, which satisfies the desired property. ∎

In order to specify the actions of a principal, we consider special transducers, so-called message-transducers, which satisfy certain properties. For this purpose, messages are interpreted as words over the finite alphabet consisting of the atomic messages as well as the letters "$\mathtt{enc}_a($", "$\mathtt{hash}($", and "$)$", that is

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

Messages considered as words over $\Sigma_{\mathcal{N}}$ have always a balanced number of opening parentheses, i.e., "$\mathtt{enc}_a($" and "$\mathtt{hash}($", 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, respectively. However, this should not lead to confusion.

Definition 9 *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* $y \in \mathcal{M}_\varepsilon$.

The first property is a condition on the "external behavior" of a message-transducer: Whenever a message-transducer gets a message as input, then the corresponding outputs are also messages (rather than arbitrary words). The second property is a rather technical condition on the "internal behavior" of a message-transducer. It is used later in Section 7 for our decidability result. However, both properties do not seem to be too restrictive. They should be satisfied for most protocols; at least they are for the transducers in the model of the recursive authentication protocol (Section 6).

An open issue is whether these properties are decidable, i.e., given a transducer over $\Sigma_{\mathcal{N}}$ does it satisfy the properties. Nevertheless, in the model of the recursive authentication protocol it is easy to see that the transducers constructed satisfy the properties.

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 actions of principals (in the sense of Definition 3) in a natural way.

In order to define all actions of a principal by a single transducer, we consider so-called 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) with $\alpha(j) = M_{\mathcal{A}_{I_j, I_{j+1}}}(I_j, I_{j+1})$ for $j < n$.

Definition 10 *A* transducer-based protocol P *is a generic protocol where the principals are defined by extended message-transducers.*

6 Modeling the Recursive Authentication Protocol

We now provide a formal description of the recursive authentication protocol in the transducer-based model. To this purpose, we first need to simplify the messages exchanged between the principals (Section 6.1). We then present the message-transducers for the agents (Section 6.2), i.e., all principals except the server, and in Section 6.3 for the server.

In what follows, let $P_0, \ldots, P_n$ be the principals participating in the recursive authentication 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, $i < n$.

6.1 Simplified Messages

In order to use message-transducers to model the principals, we simplify the messages exchanged. In the original protocol, as described in Section 2, request messages have the form

$$\mathtt{hash}_{K_{i_{l-1}}}(x_{l-1}\mathtt{hash}_{K_{i_{l-2}}}(x_{l-2}\cdots\mathtt{hash}_{K_{i_0}}(x_0)\cdots),$$

where $\mathtt{hash}_K(m) := \mathtt{hash}(Km)m$. For $l = 2$, this yields

$$\mathtt{hash}(K_{i_1}x_1\mathtt{hash}(K_{i_0}x_0)x_0)x_1\mathtt{hash}(K_{i_0}x_0)x_0.$$

The server S would check whether the hashes are taken over the correct messages, that is, the first hash in the message is really taken over K_{i_1} plus the plain text $x_1\mathtt{hash}(K_{i_0}x_0)x_0$, and in this plain text the hash is really computed from the message $K_{i_0}x_0$. For growing l, the hashes are taken over messages of growing size. Thus, the server needs unbounded memory to check whether the hash is correct: It would first read the message inside the hash, and then compare it to the plain text message. However, a transducer has only finite memory, thus cannot perform this task. To deal with this, we can proceed in two directions.

1. We fix l, and a transducer only accepts messages with the nesting depth of the hashes restricted by l.

2. A principal simply assumes, without checking, that plain text and hash match.

The first alternative allows to apply known techniques to analyze the recursive authentication protocol, since in this setting messages have a fixed and finite number of data fields to inspect. (Note that, in the recursive authentication protocol, the x_i's have a fixed and finite format.) However, some successful attacks, which may have been possible with unrestricted message size, may not work anymore.

In the second approach, the restricted computational power of principals may lead to additional successful attacks. However, the absence of an attack, guarantees that also with more powerful principals there is no attack. Therefore, we will follow this approach. In fact, the whole point of using transducers is to model principals accepting messages with an unbounded number of data fields.

Now, since transducers cannot check whether hash and plain text match, we discard the plain text altogether and only consider request messages of the following, simpler form:

$$\mathtt{hash}(K_{i_{l-1}}x_{l-1}\mathtt{hash}(K_{i_{l-2}}x_{l-2}\cdots\mathtt{hash}(K_{i_0}x_0)\cdots). \tag{1}$$

However, a principal P_i, $i < n$, given an input message of the form (1) will return a message of the form $x_l\mathtt{hash}(K_ix_l\mathtt{hash}(K_{i_{l-1}}x_{l-1}\mathtt{hash}(K_{i_{l-2}}x_{l-2}\cdots\mathtt{hash}(K_{i_0}x_0)\cdots))$, i.e., x_l is sent in plain text, because otherwise the intruder could not get hold of the request message. Note that in the recursive authentication protocol as described in Section 2, the requests (e.g., "BCN_b") are also sent in plain text (together with the message authentication code).

In what follows, we show how the principals P_i, $i \leq n$, are modeled by extended message-transducers. 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 intruders initial 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.

6.2 The Extended Message-transducer of the Agents

We now define the extended message-transducer $\mathcal{A}_i$ for P_i, $i < n$. The states of the transducer consist of three components. The first takes the values request, copy, key, and accept, indicating which step is performed: request means that P_i sends his request message. If the message consists of nested hashes, it is necessary to copy the received message. This is done in state copy. In state key, P_i waits for the response message and extracts the session keys. Then P_i proceeds to state accept. In the second component P_i stores the name of the principal who wants to share a session key with P_i. Analogously, the third component takes the name of the principal P_i has called. The latter two components have value $\perp$, if the necessary information is not available yet.

In what follows, to increase readability, a transition (p, v, w, q) is written in the following form:

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

The transducer $\mathcal{A}_i$ contains the following transitions, which are labeled with words (instead of only single letters or ε as required for message-transducers) in order to simplify the presentation:[3]

1. P_i initiates a protocol run and calls $P_{j'}$: For every $j' \leq n$,

$$(\mathsf{request}, \perp, \perp) \xrightarrow[P_iP_{j'}N_i\mathtt{hash}(K_iP_iP_{j'}N_i-)]{\varepsilon} (\mathsf{key}, \perp, P_{j'});$$

2. P_i is called by P_j and calls $P_{j'}$: For every $j < n$, $j' \leq n$, $a_0, a_1 \in \mathcal{N}$,

$$(\mathsf{request}, \perp, \perp) \xrightarrow[P_iP_{j'}N_i\mathtt{hash}(K_iP_iP_{j'}N_i\mathtt{hash}(a_0P_jP_ia_1}{\mathtt{hash}(a_0P_jP_ia_1} (\mathsf{copy}, P_j, P_{j'});$$

3. P_i copies the rest of the input message: For every $j < n$, $j' \leq n$, $a \in \mathcal{N}$,

$$(\mathsf{copy}, P_j, P_{j'}) \xrightarrow[a]{a} (\mathsf{copy}, P_j, P_{j'});$$

4. P_i concludes his request message with ")": For every $j < n$, $j' \leq n$,

$$(\mathsf{copy}, P_j, P_{j'}) \xrightarrow[)]{\varepsilon} (\mathsf{key}, P_j, P_{j'});$$

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

5. P_i, who initiated the protocol run, expects one certificate containing the session key for communication with P_j. The output message $\mathtt{enc}_a(\mathsf{secret})$ is used to check whether the intruder can get hold of a: For every $j' \leq n$ and $a \in \mathcal{N}$,

$$(\mathsf{key}, \bot, P_{j'}) \xrightarrow[\mathtt{enc}_a(\mathsf{secret})]{\mathtt{enc}_{K_i}(aP_{j'}N_i)} (\mathsf{accept}, \bot, \bot);$$

6. P_i reads the two certificates containing the session keys for communication with P_j and $P_{j'}$: For every $j < n$, $j' \leq n$, and $a_0, a_1 \in \mathcal{M}$,

$$(\mathsf{key}, P_j, P_{j'}) \xrightarrow[\mathtt{enc}_{a_0}(\mathsf{secret})\mathtt{enc}_{a_1}(\mathsf{secret})]{\mathtt{enc}_{K_i}(a_0P_{j'}N_i)\mathtt{enc}_{K_i}(a_1P_jN_i)} (\mathsf{accept}, \bot, \bot).$$

To complete the definition of the extended message-transducer, it remains to specify subsets I_0, I_1, I_2 of the state space: $I_0 := \{(\mathsf{request}, \bot, \bot)\}$, $I_1 := \{(\mathsf{key}, P_j, P_{j'}) \mid j < n, j' \leq n\}$, $I_2 := \{(\mathsf{accept}, \bot, \bot)\}$.

Since the transducer defined so far is a transducer with word-transitions, we need to turn it into one with letter transitions. For all transitions, except the ones in 2., this is done as in the proof of Lemma 8. For the transitions in 2., one first outputs $\mathtt{hash}(K_iP_iP_{j'}N_i$ (letter by letter) and then simultaneously reads and writes $\mathtt{hash}(a_0P_jP_ia_1$ (letter by letter). Thus, the outer hash is written before the inner hashes are read/written. If in 2. first the input was read and then the output (as suggested in the proof of Lemma 8), then when $\mathcal{A}_i$ performs transition 4. the closing parenthesis for the outer hash $\mathtt{hash}(K_iP_iP_{j'}N_i$ written in 2. would be written *after* the last closing parenthesis of the input. Thus, if the input is completely written, the output up to this point would not be a message. But this would violate the second property for message-transducers. Nevertheless, if in 2. the transitions are translated into transitions with letters as explained above, one can easily check that the conditions for message-transducers (cf. Definition 9) are satisfied.

6.3 The Extended Message-transducer of the Server

We define the extended message-transducer $\mathcal{A}_n$ for the server $P_n = S$. The states consist of three components. The first takes the values start, read, readpar, and accept. In the 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 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 in $\mathcal{A}_n$, again labeled with words, are specified as follows:

1. S reads the first and only request and generates the corresponding certificate: For every $a \in \mathcal{N}$,

$$(\mathsf{start}, \bot, \bot) \xrightarrow[\mathtt{enc}_{K_i}(K_{in}Sa)]{\mathtt{hash}(K_iP_iSa-)} (\mathsf{accept}, \bot, \bot);$$

2. S inspects the first two hashes, checks whether the outer hash is addresses to S, generates two certificates for P_i and one for P_j, and memorizes that possibly one more certificate must be generated for P_j: For every $i, j < n$ and $a_0, a_1 \in \mathcal{N}$,

$$(\mathsf{start}, \bot, \bot) \xrightarrow{\mathtt{hash}(K_i P_i S a_0 \mathtt{hash}(K_j P_j P_i a_1}{\mathtt{enc}_{K_i}(K_{in} S a_0)\mathtt{enc}_{K_i}(K_{ji} P_j a_0)\mathtt{enc}_{K_j}(K_{ji} P_i a_1)} (\mathsf{read}, P_j, a_1);$$

3. P_j has initiated the protocol run, and therefore, no additional certificate needs to be generated: For every $j < n$ and $a \in \mathcal{N}$,

$$(\mathsf{read}, P_j, a) \xrightarrow[\varepsilon]{-)} (\mathsf{readpar}, \bot, \bot);$$

4. The remaining certificate for P_j is generated and a new one for $P_{j'}$: For every $j, j' < n$ and $a_0, a_1 \in \mathcal{N}$,

$$(\mathsf{read}, P_j, a_0) \xrightarrow{\mathtt{hash}(K_{j'} P_{j'} P_j a_1}{\mathtt{enc}_{K_j}(K_{j'j} P_{j'} a_0)\mathtt{enc}_{K_{j'}}(K_{j'j} P_j a_1)} (\mathsf{read}, P_{j'}, a_1);$$

5. The remaining closing parentheses are read:

$$(\mathsf{readpar}, \bot, \bot) \xrightarrow[\varepsilon]{)} (\mathsf{readpar}, \bot, \bot);$$

6. S is done if the last closing parenthesis is read:

$$(\mathsf{readpar}, \bot, \bot) \xrightarrow[\varepsilon]{)} (\mathsf{accept}, \bot, \bot).$$

Since S only performs a single action, we only need to define two sets I_0 and I_1: $I_0 := \{(\mathsf{start}, \bot, \bot)\}$, $I_1 := \{(\mathsf{accept}, \bot, \bot)\}$. If $\mathcal{A}_n$ is turned into a transducer with letter transitions as in the proof of Lemma 8, it is easy to see that $(\mathcal{A}_n)_{I_0,I_1}$ is a message-transducer. In particular, the two conditions imposed on message-transducers are satisfied.

7 The Decidability Result

We show the following theorem.

Theorem 11 *For transducer-based protocols and every $h \geq 0$, h-*ATTACK *is decidable.*

Obviously, it suffices to show that there exists a decision procedure for the following problem, the so-called path problem.

PATHPROBLEM. *Given $h \geq 0$, a finite set $\mathcal{K} \subseteq \mathcal{M}_\varepsilon$, and $k \geq 0$ message-transducers $\mathcal{A}_0, \ldots, \mathcal{A}_{k-1}$ with $\mathcal{A}_i = (Q_i, \Sigma_\mathcal{N}, \{q_i^I\}, \Delta_i, \{q_i^F\})$ for $i < k$, decide whether there exist messages $m_i, m_i' \in \mathcal{M}_\varepsilon$, $i < k$, such that*

1. *$m_i \in \mathsf{d}_h(\mathcal{K} \cup \{m_0', \ldots, m_{i-1}'\})$ 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. $\mathsf{secret} \in \mathsf{d}_h(\mathcal{K} \cup \{m_0', \ldots, m_{k-1}'\})$.

We write an instance of the PATHPROBLEM as $(h, \mathcal{K}, \mathcal{A}_0, \ldots, \mathcal{A}_{k-1})$ and a solution of such an instance as a tuple $(m_0, m_0', \ldots, m_{k-1}, m_{k-1}')$ of messages. The size of instances is defined as the size of the representation for h, $\mathcal{K}$, and $\mathcal{A}_0, \ldots, A_{k-1}$.

The idea of the proof is as follows: We devise a pumping argument showing 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 $\mathcal{K}' \subseteq \mathcal{M}_\varepsilon$, it can be decided whether $m \in \mathsf{d}_h(\mathcal{K}')$.) In particular, as a "by-product" our decision procedure will yield an actual attack (if any).

To show that the length of the paths can be bounded, we proceed in three steps: First, we use a so-called solvability preserving ordering on messages, which allows to replace single messages in the initial intruder knowledge without losing solvability of the whole instance. We then consider a so-called path truncation ordering, which will indicate at which positions a path can be truncated. Finally, we show that the depth of the output of a message-transducer can be restricted, by the depth of its input. This will allow us to show that the index of the path truncation ordering is finite. These steps are now explained in more detail. However, the formal definition of the orderings are postponed to the following sections.

Preserving the solvability of instances of the path problem. For every $i \leq k$, we define a quasi-ordering[4] on messages $\preceq_i$ (the so-called solvability preserving ordering), which depends on the transducers $A_i, \ldots, A_{k-1}$, and has the following property (cf. Proposition 18): For every solvable instance $(h, \mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$ of PATHPROBLEM, every $m \in \mathcal{K}$, and $\overline{m} \in \mathcal{M}_\varepsilon$ with $m \preceq_i \overline{m}$, the instance $(h, (\mathcal{K} \setminus \{m\}) \cup \{\overline{m}\}, \mathcal{A}_i, \ldots, A_{k-1})$ is solvable as well.

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 $\overline{m}_i \in \mathsf{d}_h(\mathcal{K} \cup \{m_0', \ldots, m_{i-1}'\})$ 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 $\mathcal{K} \cup \{m_0', \ldots, m_{i-1}', \overline{m}_i'\}$ instead of $\mathcal{K} \cup \{m_0', \ldots, m_{i-1}', m_i'\}$. Thus, using Proposition 18, there still exists a solution for the rest of the instance, i.e., $(h, \mathcal{K} \cup \{m_0', \ldots, m_{i-1}', \overline{m}_i'\}, \mathcal{A}_{i+1}, \ldots, A_{k-1})$.

Consequently, it remains to find criteria for truncating long paths in this way. It is rather straightforward to satisfy the condition on the input label ($\overline{m}_i \in \mathsf{d}_h(\mathcal{K} \cup \{m_0', \ldots, m_{i-1}'\})$). The involved part is the condition on the output label ($m_i' \preceq_{i+1} \overline{m}_i'$). In what follows, we will therefore focus on the latter condition.

Truncating paths. We extend $\preceq_i$ to a quasi-ordering $\preceq_i^l$ (the path truncation ordering) on so-called left half-messages. Left half-messages are prefixes of messages

[4]Recall that a quasi-ordering is a reflexive and transitive ordering.

(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 (cf. Proposition 22): 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 $\alpha_x \equiv_i^l \alpha_y$. Then, by Proposition 22, it follows $m_i' = \alpha_y\gamma_y \equiv_i \alpha_x\gamma_y$, where $\alpha_x\gamma_y$ is the output label of the path obtained by cutting out the subpath in π between x and y.[5] Thus, $\equiv_i^l$ provides us with the desired criterion for "safely" (in the sense of 1.) shortening paths. 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 $\equiv_i^l$ (i.e., the number of equivalence classes modulo $\equiv_i^l$ on left half-messages of level l) can be bounded in the size of the problem instance. To this purpose, the following is shown.

Bounding the depth of the output of message-transducers. Let π be a path in $\mathcal{A}_i$ from q_i^I to q_i^F or a strict path in $\mathcal{A}_i$, and x be a position in π such that α_x is the input label of π up to position x and β_x is the output label of π up to x. Then, the level of β_x can be bounded by a polynomial in the level of α_x and the number of states of $\mathcal{A}_i$ (cf. Proposition 23).

With this, it is not hard to show the bound on the index of $\equiv_i^l$ (cf. Proposition 30). Also, since in an h-attack the intruder can only produce messages of depth bounded in the size of the problem instance, we know that the depth of m_0 is bounded (Observation 2). Thus, using Proposition 23, the depth of m_0' is bounded as well, and by induction, the depth of all messages $m_1, m_1', \ldots, m_{k-1}, m_{k-1}'$. Therefore, the l in 2. (the level of the half-messages α_x, α_y, γ_x, γ_y) is bounded in the size of the problem instance.

Following these steps, we now provide a detailed proof. We will define the different orderings and prove the needed properties. In Section 7.5 everything will be put together to show decidability of the path problem.

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

[5] One little technical problem is that $\alpha_x\gamma_y$ does not need to be a message since it may contain a word of the form $\mathtt{enc}_a()$, 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.

7.1 The Solvability Preserving Ordering $\preceq_i$

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

Definition 12 *For messages $m, m' \in \mathcal{M}_\varepsilon$, we define $m \preceq m'$ iff for all $N \subseteq \mathcal{N}$: $\mathsf{an}_{\mathcal{N}}(m, N) \subseteq \mathsf{an}_{\mathcal{N}}(m', N)$.*

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 $a, a', a'' \in \mathcal{N}$, $\mathtt{enc}_a(\mathtt{enc}_{a'}(a'')) \preceq a'\mathtt{enc}_{a'}(\mathtt{enc}_a(a''))$. Obviously, $\preceq$ is a quasi-ordering, i.e., it is reflexive and transitive.

The relation $\preceq_i$, $i \leq k$, depends on the message-transducers $\mathcal{A}_i, \ldots, \mathcal{A}_{k-1}$ and is defined inductively. The definition also needs the ordering $\sqsubseteq_i$. To understand the definition of $\preceq_i$ recall that we want to guarantee that if $(h, \mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$ has a solution, say $(m_i, m_i', \ldots, m_{k-1}, m_{k-1}')$, then so has $(h, \mathcal{K} \setminus \{m\} \cup \{m'\}, \mathcal{A}_i, \ldots, A_{k-1})$. The message m_i may have been built up 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 $\preceq_{i+1}$ (1.). Finally, in our proofs we will use that $\preceq_i$ is closed under substitution. With condition 3 this can be shown.

Definition 13 *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 $x = \mathtt{enc}_a(z)$ for some $a \in \mathcal{N}$ and $z \in \mathcal{M}$, there exists $x' := \mathtt{enc}_{a'}(z') \in \mathsf{an}(m', N)$ 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 $y \preceq_{i+1} y'$.

The following lemma is proved by a simple induction on $i \leq k$.

Lemma 14 *For every $i \leq k$, $\preceq_i$ is a quasi-ordering.*

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 $V = \{v_0, \ldots, v_{n-1}\}$ of variables and a subset $T \subseteq \mathcal{T}_\varepsilon(V)$ of terms over V, we define $\mathsf{an}_c(T)$ to be the closure of T under decomposition, $\mathsf{an}_e(T)$ to be the

closure of T under decryption, and $\mathsf{an}_{cec}(T)$ the closure of T under decomposition, encryption, and decomposition (in this order), where "c" stands for composition and "e" for encryption. Formally,

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

It is easy to see that

$$\mathsf{an}(T) = \bigcup_{i \geq 0} \mathsf{an}^i_{cec}(T), \tag{2}$$

where $\mathsf{an}^0_{cec}(T) := T$ and $\mathsf{an}^{i+1}_{cec}(T) := \mathsf{an}_{cec}(\mathsf{an}^i_{cec}(T))$.

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

Given a term t and $N \subseteq \mathcal{N}$, we say that a subterm t' of t is *N-accessible*[6] if

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

Lemma 15 *Let $x_0, \ldots, x_{n-1}, x'_0, \ldots, x'_{n-1} \in \mathcal{M}$ be messages and $t(v_0, \ldots, v_{n-1})$ be a term. If $x_i \preceq x'_i$ for all $i < n$, then $t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}] \preceq t[v_0/x'_0, \ldots, v_{n-1}/x'_{n-1}]$.*

Proof. Define $m := t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$ and $m' := t[v_0/x'_0, \ldots, v_{n-1}/x'_{n-1}]$. Because of (2) it suffices to show

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

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

Assume $i = 0$ and $a \in (\{m\} \cup N) \cap \mathcal{N}$. If $a \in N$, nothing is 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 $a \in \mathsf{an}_{\mathcal{N}}(m', N)$. In the latter case, $x_i \preceq x'_i$ implies $x'_i = a$, and thus, $a \in \mathsf{an}_{\mathcal{N}}(m', N)$.

Assume $i > 0$ and $a \in \mathsf{an}_{cec}(\mathsf{an}^i_{cec}(m, N)) \cap \mathcal{N}$. If $a \in \mathsf{an}^i_{cec}(m, N)$, the induction hypothesis yields $a \in \mathsf{an}_{\mathcal{N}}(m', N)$. Otherwise, there must exist $x \in \mathsf{an}^i_{cec}(m, N)$, a message z, and $b \in \mathsf{an}^i_{cec}(m, N) \cap \mathcal{N}$ with $x = \mathtt{enc}_b(z)$ and $a \in \mathsf{an}_c(z)$. We distinguish two cases.

i) There exists a term t' such that $\mathtt{enc}_b(t')$ is a subterm of t, $z = t'[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$, and t' is $(\mathsf{an}^i_{cec,\mathcal{N}}(m, N))$-accessible in t. Thus, by the induction hypothesis, t' is $\mathsf{an}_{\mathcal{N}}(m', N)$-accessible in t. Consequently, if $a \in \mathsf{an}_c(t')$, then $a \in \mathsf{an}_{\mathcal{N}}(m', N)$. Otherwise, there must exist $i < n$, $t_0, t_1 \in \mathcal{T}_\varepsilon(V)$ such that $t' = t_0 v_i t_1$, and $a \in \mathsf{an}_c(\{x_i\})$. Because $x_i \preceq x'_i$, it follows $a \in \mathsf{an}(x'_i)$, and therefore, $a \in \mathsf{an}_{\mathcal{N}}(m', N)$.

ii) There exists $i < n$ such that x, and thus z, is a submessage of x_i. Let $N' := \mathsf{an}^i_{cec,\mathcal{N}}(m, N)$. By induction hypothesis $N' \subseteq \mathsf{an}_{\mathcal{N}}(m', N)$. We know that z and x_i

[6] This notion was already defined in [1]

are N'-accessible in m. Thus, x'_i is N'-accessible in m', and therefore also $\mathsf{an}_{\mathcal{N}}(m', N)$-accessible. Now $x_i \preceq x'_i$ implies $\mathsf{an}(x_i, N') \subseteq \mathsf{an}(x'_i, N')$, and we know $a \in \mathsf{an}(x_i, N')$. Thus, $a \in \mathsf{an}(x'_i, N')$. With $N' \subseteq \mathsf{an}_{\mathcal{N}}(m', N)$ this yields $a \in \mathsf{an}(x'_i, \mathsf{an}_{\mathcal{N}}(m', N))$. Finally, since x'_i is $\mathsf{an}_{\mathcal{N}}(m', N)$-accessible in m', we obtain $a \in \mathsf{an}_{\mathcal{N}}(m', N)$. ∎

We generalize Lemma 15 to the solvability preserving ordering $\preceq_i$.

Lemma 16 *Let $x_0, \ldots, x_{n-1}, x'_0, \ldots, x'_{n-1} \in \mathcal{M}_\varepsilon$, $i \leq k$, and $t(v_0, \ldots, v_{n-1})$ be a term, where every variable v_i occurs at most once in t. If $x_j \preceq_i x'_j$ for all $j < n$, then $t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}] \preceq_i t[v_0/x'_0, \ldots, v_{n-1}/x'_{n-1}]$.*

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.

The proof is by induction on i. If $k = i$, the statement follows immediately from Lemma 15. Define $m := t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$ and $m' := t[v_0/x'_0, \ldots, v_{n-1}/x'_{n-1}]$. Assume $i < k$. Following Definition 13, we show $m \preceq_i m'$.

1. From $x_j \preceq_i x'_j$, $j < n$, it follows $x_j \preceq_{i+1} x'_j$. Thus, by the induction hypothesis, $m \preceq_{i+1} m'$.

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

   i) There exists a subterm t' of t such that $x = t'[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$ and $t' \notin \{v_0, \ldots, v_{n-1}\}$. Let $x' := t'[v_0/x'_0, \ldots, v_{n-1}/x'_{n-1}]$. We know that t' is $\mathsf{an}_{\mathcal{N}}(m, N)$-accessible in t. Since $m \preceq m'$, t' is also $\mathsf{an}_{\mathcal{N}}(m', N)$-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 $\mathtt{enc}_a(t'')$ for some term t'' over $\{v_0, \ldots, v_{n-1}\}$. Thus, x' has the form $\mathtt{enc}_a(z')$ for some message z'. It remains to show that $x \sqsubseteq_i x'$.

   Let $p, q \in Q_i$, $y \in \mathcal{M}_\varepsilon$ with $\pi := p(x, y)q \in_s A_i$. We know that x is of the form $t'[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$. Thus, if an 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 term $t''(v_0, \ldots, v_{n-1})$, where every variable occurs at most once, such that $y = t''[v_0/y_0, \ldots, v_{n-1}/y_{n-1}]$.

   Because $x_j \preceq_i x'_j$, there exists $y'_j \in \mathcal{M}_\varepsilon$ with $p_j(x'_j, y'_j), p'_j \in_s \mathcal{A}_i$ and $y_j \preceq_{i+1} y'_j$. Set $y' := t''[v_0/y'_0, \ldots, v_{n-1}/y'_{n-1}]$. By induction hypothesis, $y \preceq_{i+1} y'$. Furthermore, replacing in π the subpaths $p_j(x_j, y_j)p'_j$ by $p_j(x'_j, y'_j), p'_j$ shows that $p(x', y')q \in_s A_i$.

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

3. Similar to 2.,i), one shows $m \sqsubseteq_i m'$. ∎

The main property of the solvability preserving ordering. 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 17 *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. The proof is very similar to the one for Lemma 15. We show $\mathsf{an}^i_{cec,\mathcal{N}}(m, \mathcal{K}) \subseteq \mathsf{an}_\mathcal{N}(m', \mathcal{K})$ for every $i \geq 0$ by induction on i.

Assume $i = 0$ and $a \in (\{m\} \cup \mathcal{K}) \cap \mathcal{N}$. 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, $a \in \mathsf{an}_\mathcal{N}(m', \mathcal{K})$.

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

i) The messages x and z are submessage of some message x' in $\mathcal{K}$. In particular, x and z are $\mathsf{an}^i_{cec,\mathcal{N}}(m, \mathcal{K})$-accessible in x', and thus, x and z are $\mathsf{an}_\mathcal{N}(m', \mathcal{K})$-accessible in x'. Consequently, $a \in \mathsf{an}_\mathcal{N}(m', \mathcal{K})$.

ii) The messages x and z are submessages of m. Let $N := \mathsf{an}^i_{cec,\mathcal{N}}(m, \mathcal{K})$. By induction hypothesis $N \subseteq \mathsf{an}_\mathcal{N}(m', \mathcal{K})$. We know $a \in \mathsf{an}_\mathcal{N}(m, N)$, and $m \preceq m'$ implies $\mathsf{an}_\mathcal{N}(m, N) \subseteq \mathsf{an}_\mathcal{N}(m', N)$. Thus, $a \in \mathsf{an}_\mathcal{N}(m', N) \subseteq \mathsf{an}_\mathcal{N}(m', \mathcal{K})$. ∎

Using Lemma 17 and 16, we prove the main statement of this subsection.

Proposition 18 *Assume that $(h, \mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$, $i \leq k$, is 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 $(h, \overline{\mathcal{K}}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$ 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 nothing has to be shown. Therefore, we assume that $m \neq \varepsilon$, and thus, $\overline{m} \neq \varepsilon$. The induction basis, $i = k$, immediately follows from Lemma 17.

Now assume $i < k$. Define $N := \mathsf{an}_\mathcal{N}(\mathcal{K})$ and $M := \{\mathsf{enc}_a(z) \in \mathsf{an}(m, N) \mid$ there exist $z \in \mathcal{M}$ and $a \in \mathcal{N}\}$. Let $(m_i, m'_i, \ldots, m_{k-1}, m'_{k-1})$ be a solution of $(h, \mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$. Thus, $m_i \in \mathsf{d}_h(\mathcal{K})$. If in an derivation for m_i, a message of the form $x\mathsf{enc}_a(z)y \in \mathsf{an}(m, N)$ is used to construct m_i, we will w.l.o.g. assume that the single messages $x, \mathsf{enc}_a(z), y$ are used, since all three messages belong to $\mathsf{an}(m, N)$. Let n be the number of times a message in M was used to derive m_i 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. (Due to the assumption on derivations made before, every occurrence of some message in M is taken into account.) Also, let $v_0, \ldots, v_{n-1}$ be pairwise distinct variables. Then, there exists a term t over $\{v_0, \ldots, v_{n-1}\}$ with depth at most h such that $m_i = t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$. 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 $\overline{m}_i := t[v_0/\overline{x}_0, \ldots, v_{n-1}/\overline{x}_{n-1}]$. Since, by Lemma 17, $N \subseteq \mathsf{an}(\overline{\mathcal{K}})$, we can conclude $\overline{x}_j \in \mathsf{an}(\overline{\mathcal{K}})$, and it follows $\overline{m}_i \in \mathsf{d}_h(\overline{\mathcal{K}})$, since the derivation of $\overline{m}_i$ from $\overline{\mathcal{K}}$ coincides with the one for m_i from $\mathcal{K}$ except that the x_j's are replaced by $\overline{x}_j$.

In $\pi = q^I_i(m_i, m'_i)q^F_i \in \mathcal{A}_i$, there exist subpaths of the form $p_j(x_j, y_j)p_{j+1} \in_s \mathcal{A}_i$, for every $j < n$. By the properties of message-transducers, we know that $y_j \in \mathcal{M}_\varepsilon$,

and there exists a term t' over $\{v_0, \ldots, v_{n-1}\}$, where every v_j, $j < n$, occurs exactly once in t', with $m'_i = t'[v_0/y_0, \ldots, v_{n-1}/y_{n-1}]$. Since $x_j \sqsubseteq_i \overline{x}_j$, there exists $\overline{y}_j$ with $p_j(\overline{x}_j, \overline{y}_j)p_{j+1} \in_s \mathcal{A}_i$ and $y_j \preceq_{i+1} \overline{y}_j$. Define $\overline{m}'_i := t'[v_0/\overline{y}_0, \ldots, v_{n-1}/\overline{y}_{n-1}]$.

If we replace in π every subpath $p_j(x_j, y_j)p_{j+1} \in_s \mathcal{A}_i$ by $p_j(\overline{x}_j, \overline{y}_j)p_{j+1} \in_s \mathcal{A}_i$, we obtain $q_i^I(\overline{m}_i, \overline{m}'_i)q_i^F \in \mathcal{A}_i$.

By Lemma 16, we have $m'_i \preceq_{i+1} \overline{m}'_i$. Thus, since $(h, \mathcal{K} \cup \{m'_i\}, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_{k-1})$ has a solution, by induction hypothesis, the instance $(h, \mathcal{K} \cup \{\overline{m}'_i\}, \mathcal{A}_{i+1}, \ldots, \mathcal{A}_{k-1})$ is also solvable. Finally, since $m \preceq_i \overline{m}$ implies $m \preceq_{i+1} \overline{m}$, the induction hypothesis also yields a solution for $(h, \overline{\mathcal{K}} \cup \{\overline{m}'_i\}, \mathcal{A}_{i+1}, \ldots, A_{k-1})$, and together with $\overline{m}_i$ and $\overline{m}'_i$, this is a solution for $(h, \overline{\mathcal{K}}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$. ∎

7.2 The Path Truncation Ordering $\preceq_i^l$

We now extend $\preceq_i$ to the path truncation ordering $\preceq_i^l$ on half-messages.

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. Analogously, a *right half-message* is a suffix of a message.

For a left half-message α, the *level* $l(\alpha)$ *of* α is defined as the number of symbols "$\mathtt{enc}_a($", 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 "$\mathtt{enc}_a($", for some $a \in \mathcal{N}$.

If α is a left half-message, then there exist unique messages $x_0, \ldots, x_{l(\alpha)} \in \mathcal{M}_\varepsilon$, and $a_1, \ldots, a_{l(\alpha)-1} \in \mathcal{N}$ such that

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

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

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

for $1 \leq j \leq l(\alpha)$. Note that $l(\alpha_j) = j$. 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 19 *Let $l \geq 0$ and α, α' be non-empty left half-messages of level l, i.e., $\alpha \neq \varepsilon$, $\alpha' \neq \varepsilon$, 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 quasi-ordering. 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, 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.

Definition 20 *For every $l \geq 0$ and $i \leq k$, 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) $\alpha \neq \varepsilon$, $\alpha' \neq \varepsilon$, $\alpha \preceq^l \alpha'$, 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 $1 \leq j \leq l$;*
4. *if $l \geq 1$ and $x, x' \in \mathcal{M}_\varepsilon$ with $\alpha = x\alpha_l$ and $\alpha' = x'\alpha'_l$, then $x \sqsubseteq_i x'$.*

For $i < k$, $l \geq 0$, left half-messages α, α' of level l, we define $\alpha \sqsubseteq_i^l \alpha'$ iff i) $\alpha = \alpha' = \varepsilon$, or ii) $\alpha \neq \varepsilon$, $\alpha' \neq \varepsilon$, 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 $\beta \preceq_{i+1}^{l(\beta)} \beta'$.

We now show, by induction on i, that $\preceq_i^l$ is compatible with right concatenation of right half-messages. The case $i = k$ is shown in the following lemma.

Lemma 21 *Let α, α' be left half-messages of level $l \geq 0$. 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, then the lemma follows from Lemma 15, when we set $t := v_0 v_1$ and consider $t[v_0/\alpha, v_1/\gamma]$ and $t[v_0/\alpha', v_1/\gamma]$. In what follows we assume $l > 0$.

We need to show that $\mathsf{an}_\mathcal{N}(\alpha\gamma, N) \subseteq \mathsf{an}_\mathcal{N}(\alpha'\gamma, N)$ 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 $N \subseteq \mathcal{N}$,

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

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

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

(Note that the corresponding closing parenthesis to "$\mathtt{enc}_{a_j}($" 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, $F(N) \subseteq F'(N)$. Using similar techniques as in the proof of Lemma 15, it is rather straightforward to show

$$\begin{aligned} \mathsf{an}_\mathcal{N}(\alpha\gamma, N) &= \mathsf{lfp}_N(F) := \bigcup_{j \geq 0} F^j(N), \text{ and} \\ \mathsf{an}_\mathcal{N}(\alpha'\gamma, N) &= \mathsf{lfp}_N(F'). \end{aligned}$$

From this the lemma follows. ∎

We can now proof the main proposition of this subsection.

Proposition 22 *Let α, α' be left half-messages of level $l \geq 0$ and let $i \leq k$. 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 21.

Now assume $i < k$. If $l = 0$, then α, α', and γ are messages, and $\alpha\gamma \preceq_i \alpha'\gamma$ follows from Lemma 16 with $t = v_0 v_1$. Therefore, we may assume that $l > 0$. To prove $\alpha\gamma \preceq_i \alpha'\gamma$, we must show the conditions in Definition 13.

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

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

   (a) x is a submessage of γ. According to Lemma 21, $\alpha\gamma \preceq \alpha'\gamma$. Thus, $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$. Using $p(\alpha) = p(\alpha')$, it follows $x \in \mathsf{an}(\alpha'\gamma, N)$, and we can simply choose $x' := x$.

   (b) x is a submessage of α. It follows $x \in \mathsf{an}(\alpha^*, \mathsf{an}_{\mathcal{N}}(\alpha\gamma, N))$. Since $\alpha^* \preceq_i \alpha'^*$, there exists $x' \in \mathsf{an}(\alpha'^*, \mathsf{an}(\alpha\gamma, N))$ such that x' is of the form $\mathtt{enc}_b(z')$ for some message z' and $b \in \mathcal{N}$ and $x \sqsubseteq_i x'$. Finally, because $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$ (Lemma 21), it follows $x' \in \mathsf{an}(\alpha'^*, \mathsf{an}(\alpha'\gamma, N))$, and thus, $x' \in \mathsf{an}(\alpha'\gamma, N)$.

   (c) 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 $l(\gamma') = l(\alpha_j)$. Define $x' := \alpha'_j\gamma'$. Obviously, x' is a message of the form $\mathtt{enc}_a(z')$ for some message z' and $a \in \mathcal{N}$. Since $\mathsf{an}_{\mathcal{N}}(\alpha\gamma, N) \subseteq \mathsf{an}_{\mathcal{N}}(\alpha'\gamma, N)$ (Lemma 21) and $p(\alpha) = p(\alpha')$ it follows $x' \in \mathsf{an}(\alpha'\gamma, N)$. We need to show $x \sqsubseteq_i x'$.

   Let $p, q \in Q_i$, $y \in \mathcal{M}_\varepsilon$ such that $p(x, y)q \in_s \mathcal{A}_i$. There exists $p' \in Q_i$, and 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 $\alpha_j \sqsubseteq_i^{l(\alpha_j)} \alpha'_j$, and consequently, there exists a left half-message β'' with $l(\beta) = l(\beta'')$, $p(\alpha'_j, \beta')p' \in_s \mathcal{A}_i$, 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, $y = \beta\beta' \preceq_{i+1} \beta''\beta' = y'$.

3. 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. 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 with $p(x, y_0)p_0 \in_s \mathcal{A}_i$, $p_0(\varepsilon, y_1)p_1 \in \mathcal{A}_i$, $p_1(\alpha_l, \beta)p_2 \in_s \mathcal{A}_i$, 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-transducer, 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. Particularly, 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 $x \sqsubseteq_i x'$. Thus, there exists $y'_0 \in \mathcal{M}_\varepsilon$ with $p(x', y'_0), p_0 \in_s \mathcal{A}_i$ and $y_0 \preceq_{i+1} y'_0$. Therefore, replacing in the path $p(x, y)q$ the subpath $p(x, y_0)p_0$ by $p(x', y'_0), p_0$ and $p_1(\alpha_l, \beta)p_2$ 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 16, with $t = v_0 v_1 v_2$ we obtain $y = t[v_0/y_0, v_1/y_1, v_2/\beta\beta'] \preceq_{i+1} t[v_0/y'_0, v_1/y_1, v_2/\beta''\beta'] = y'$.

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

7.3 Bounding the Depth of Output Labels

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

$$q_0(a_0, b_0)q_1(a_1, b_1)\cdots(a_{r-1}, b_{r-1})q_r \tag{3}$$

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.

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.

Proposition 23 *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 (3) 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 $l'_\pi(i) \leq (n^2 \cdot (2n + 1) + 1) \cdot (l_\pi(i) + 1)$ for every $i \leq r$.*

We define

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

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

Corollary 24 *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 $p, q \in Q$:* $\mathsf{depth}(m') \leq \mathsf{depth}(\mathcal{A}) \cdot (\mathsf{depth}(m) + 1)$.

The proof uses that the length of paths in $\mathcal{A}$ can be restricted by $\mathsf{depth}(\mathcal{A})$. To show this we cannot simple use the usual pumping argument on finite automata since if we truncate a path, we want 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 $\mathtt{enc}_a(\cdots \mathtt{enc}_b(\cdot)\cdots)$ is replaced by $\mathtt{enc}_b(\cdot)$. 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 $\mathtt{enc}_a()$, which is not a message. The following lemma, shows how to solve this problem. For a message $m = c_0 \cdots c_{r-1} \in \mathcal{M}$ with $c_i \in \Sigma_{\mathcal{N}}$, $i < k$, let $l_m(i) := l(c_0 \cdots c_{i-1})$ for $i \leq r$.

Lemma 25 *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. *Let $0 < i < j < r$ with $l_m(i) = l_m(j)$. Then, $c_{i-1} \neq \mathtt{enc}_a($, for any $a \in \mathcal{N}$, or $c_j \neq)$ iff $c_0 \cdots c_{i-1} c_j \cdots c_{r-1} \in \mathcal{M}_\varepsilon$.*
3. *If $0 < i_0 < i_1 < i_2 < r$ 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 easy to see. For the second statement, we note that the condition $c_{i-1} \neq \mathtt{enc}_a($, for any $a \in \mathcal{N}$, or $c_j \neq)$ guarantees that $c_{i-1}c_j$ is not of the form $\mathtt{enc}_b()$, for some $b \in \mathcal{N}$. Having ruled out this possibility, it is not hard 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 $\mathtt{enc}_b()$, it follows $c_{i-1} \neq \mathtt{enc}_a($, for any $a \in \mathcal{N}$, or $c_j \neq)$.

For the third statement, 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 2. it follows: $c_{i_0} = \mathtt{enc}_a($, for some $a \in \mathcal{N}$, $c_{i_1} =)$, $c_{i_1-1} = \mathtt{enc}_b($, for some $b \in \mathcal{N}$, and $c_{i_2} =)$. But then c contains as a submessage $c_{i_1-1}c_{i_1} = \mathtt{enc}_b()$, 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 26 *Let π be a path of the form (3) in a message-transducer $\mathcal{A}$, 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 $q_{i_0} = q_{j_0}$, $q_{j_1} = q_{i_1}$, $a_{i_0} = \mathtt{enc}_a($, for some $a \in \mathcal{N}$, $a_{i_1-1} =)$ (the corresponding closing parenthesis to a_{i_0}), and analogously, $a_{j_0} = \mathtt{enc}_b($, 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, for $l_\pi(i)$, as defined above, we know $l_\pi(i) \leq n^2$ 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 $l_\pi(i_0) = l_\pi(i_1) = l_\pi(i_2)$. It follows that for $j \in \{0,1\}$ the path π'_j given as

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

is a path in $\mathcal{A}$ from q_0 to q_r. Lemma 25 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. ∎

Proof of Proposition 23. Let π be a path as given in Proposition 23, and assume 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.

$l_\pi(i) > 0$. Let $j > i$ be minimal with $l_\pi(j) = l_\pi(i) - 1$. From the minimality of $l_\pi(i)$ it follows $l'_\pi(j) \leq \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(j)+1)$. But then, there must exist $> \mathsf{depth}(\mathcal{A})$ many positions s with $i \leq s \leq j-1$ and $b_s =)$. (Otherwise, $l'_\pi(j) = l'_\pi(i) - g$ for some $g \leq \mathsf{depth}(\mathcal{A})$. Thus, $l'_\pi(j) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(i)+1) - \mathsf{depth}(\mathcal{A}) = \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(j)+1)$, 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 26 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 $l_{\overline{\pi}}(\overline{j}) < l_{\overline{\pi}}(i)$, since $< \mathsf{depth}(\mathcal{A})$ parenthesis have been closed between position i and $\overline{j}-1$ in $\overline{\pi}$, i.e., $\leq \mathsf{depth}(\mathcal{A})$ parenthesis 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.)

$l_\pi(i) = 0$. Let π' denote the path from q_i to q_r with input label $a_i \cdots a_{r-1}$ and output label $b_i \cdots b_{r-1}$. Since $l'_\pi(i) > \mathsf{depth}(\mathcal{A}) \cdot (l_\pi(i)+1)$, the output label must contain $> \mathsf{depth}(\mathcal{A})$ closing parentheses. However, according to Lemma 26, 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 a input

label distinct from ε, then, according to Lemma 26, 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 at least two closing parenthesis are missing.

7.4 The Index of $\preceq_i^l$

If $\leq$ is a quasi-ordering 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 $[a]_{\equiv} := \{b \mid a \equiv b\}$. The *index* $I(\equiv)$ of $\equiv$ is the number of different equivalence classes over S.

Let $\equiv$, $\equiv_i$, $=_i$, $\equiv^l$, $\equiv_i^l$, and $=_i^l$ denote the equivalence relations corresponding to $\preceq$, $\preceq_i$, $\sqsubseteq_i$, $\preceq^l$, $\preceq_i^l$, and $\sqsubseteq_i^l$.

We show that the index of $\equiv_i^l$ is finite. To this end, we first consider $\equiv$ and $\equiv_i$. The proof of the following lemma is straightforward.

Lemma 27 *The index of $\equiv$ is finite. More precisely,*

$$I(\equiv) \in \mathcal{O}(2^{2^{|\mathcal{N}|} \cdot |\mathcal{N}|})$$

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

Lemma 28 *The index of $\equiv_k$, and for every $i < k$, the index of $\equiv_i$ and $=_i$ is finite, and can be bounded as follows:*

$$\begin{aligned} I(=_i) &\in \mathcal{O}(2^{I(\equiv_{i+1}) \cdot 2^{|Q_i|^2}}) \\ I(\equiv_i) &\in \mathcal{O}(I(\equiv_{i+1}) \cdot 2^{I(=_i) \cdot 2^{|\mathcal{N}|}} \cdot I(=_i)) \end{aligned}$$

Proof. The proof is by induction on i. For $i = k$, Lemma 27 shows that the index of $\equiv_k$ is finite. Assume that $i < k$ and the index of $\equiv_{i+1}$ is finite. We first show that the index of $=_i$ is finite and from this conclude that $\equiv_i$ has finite index.

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 $p, q \in Q_i$.

To prove that $=_i^t$ has finite index, consider the mapping f_i^t which takes every tuple (x, y) to the tuple $([y]_{\equiv_{i+1}}, \{(p, q) \mid p(x, y)q \in_s \mathcal{A}_i\})$. Since $\equiv_{i+1}$ has a finite index, it follows that the range of f_i^t is finite. Moreover, it is easy to see that $f_i^t(x, y) = f^t(x', y')$ implies $(x, y) =_i^t (x', y')$. From this, it immediately follows that $=_i^t$ has finite index, and that

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

To show that $=_i$ has finite index, define $M_{i,x} := \{[(x,y)]_{=_i^t} \mid y \in \mathcal{M}_\varepsilon\}$. Clearly, $M_{i,x}$ is a finite set, and there are only finitely many different sets $M_{i,x}$. Together with the following claim, this implies that $=_i$ has finite index.

Claim I. For messages $x, x' \in \mathcal{M}$, $M_{i,x} = M_{i,x'}$ implies $x =_i x'$.

Proof of the claim. Assume $M_{i,x} = M_{i,x'}$. We show $x \sqsubseteq_i x'$; $x' \sqsubseteq_i x$ follows by symmetry. Let $p, q \in Q_i$ and $y \in \mathcal{M}_\varepsilon$ with $p(x,y)q \in_s \mathcal{A}_i$. We know $[(x,y)]_{=_i^t} \in M_{i,x} = M_{i,x'}$. Thus, there exists $y' \in \mathcal{M}_\varepsilon$ such that $[(x,y)]_{=_i^t} = [(x',y')]_{=_i^t}$. Consequently, by definition of $=_i^t$, $y \equiv_{i+1} y'$ and $p(x',y')q \in_s \mathcal{A}_i$. This shows $x \sqsubseteq_i x'$, and concludes the proof of the claim.

From this it immediately follows that

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

We now show that the index of $\equiv_i$ is finite. Let $M_{i,m,N} := \{[x]_{=_i} \mid x \in \mathsf{an}(m,N)$ and $x = \mathtt{enc}_a(z)$ for $z \in \mathcal{M}$ and $a \in \mathcal{N}\}$, and f_i be a mapping that takes every message $m \in \mathcal{M}$ to $([m]_{\equiv_{i+1}}, (M_{i,m,N} \mid N \subseteq \mathcal{N}), [m]_{=_i})$. We know that the index of $\equiv_{i+1}$ and $=_i$ is finite. Thus, $M_{i,m,N}$ is finite and there are only finitely many different sets $M_{i,m,N}$. Also, recall that $\mathcal{N}$ is a finite set. As a consequence, the range of f_i is finite. Together with the following claim, we can conclude that $\equiv_i$ has finite index.

Claim II. For messages $m, m' \in \mathcal{M}$, $f_i(m) = f_i(m')$ implies $m \equiv_i m'$.

Proof of the claim. Assume $f_i(m) = f_i(m')$. We show $m \preceq_i m'$; $m' \preceq_i m$ follows by symmetry.

1. From $f_i(m) = f_i(m')$, we immediately obtain $m \preceq_{i+1} m'$, and in particular, $m \preceq m'$.
2. Let $N \subseteq \mathcal{N}$, $x \in \mathsf{an}(m,N)$ with $x = \mathtt{enc}_a(z)$ for some $z \in \mathcal{M}$ and $a \in \mathcal{N}$. Thus, $[x]_{=_i} \in M_{i,m,N} = M_{i,m',N}$. Consequently, there exists $x' \in \mathsf{an}(m',N)$ with $x' = \mathtt{enc}_b(z')$ for some $z' \in \mathcal{M}$, $b \in \mathcal{N}$, and $[x']_{=_i} = [x]_{=_i}$. In particular, $x \sqsubseteq_i x'$.
3. From $f_i(m) = f_i(m')$, $m =_i m'$ follows immediately, and thus, $m \sqsubseteq_i m'$.

This concludes the proof of the claim.

As an immediate consequence, we obtain that

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

∎

We prove that $\equiv_i^l$ has finite index by induction on i. The base case follows from the following lemma.

Lemma 29 *For every $l \geq 0$, the index of $\equiv^l$ is finite, and can be bounded as follows:*

$$I(\equiv^l) \in \mathcal{O}(I(\equiv) \cdot |\mathcal{N}|^l)$$

PROOF. Let f_l be a mapping that takes a left half-message α of level l to the tuple $([\alpha^*]_{\equiv}, p(\alpha))$. Since $\equiv$ has finite index and there are only a finite number of words over $\mathcal{N}$ of length l, the range of f_l is finite, namely, $I(\equiv) \cdot |\mathcal{N}|^l$. Moreover, it is easy to see that $f_l(\alpha) = f_l(\alpha')$ implies $\alpha \equiv^l \alpha'$ for all left half-messages α and α' of level l. ∎

The following proposition is the main statement of this subsection. The proof is very similar to the one for Lemma 28. However, it requires Proposition 23.

Proposition 30 *For all $l \geq 0$, the index of $\equiv_k^l$, and for every $i < k$, the index of $\equiv_i^l$ and $=_i^l$ is finite, and can be bounded as follows:*

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

PROOF. The proof is by induction on i. For $i = k$, Lemma 29 shows that the index of $\equiv_k^l$ is finite. Assume that $i < k$ and the index of $\equiv_{i+1}^l$ is finite. We first show that the index of $=_i^l$ is finite and from this conclude that $\equiv_i^l$ has finite index.

For every $l, l' \geq 0$, we introduce a new equivalence relation on tuples (α, β) with left half-messages α and β. More precisely, 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 Lemma 28, one shows that $=_i^{l,l'}$ has finite index, and that

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

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

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

and the tuple

$$M_{i,\alpha}^{l} := (M_{i,\alpha}^{l,l'} \mid l' \leq \mathsf{depth}(\mathcal{A}_i) \cdot (l+1)).$$

Clearly, for fixed l, l', and i the number of different sets $M_{i,\alpha}^{l,l'}$ is finite. Consequently, the set of tuples $M_{i,\alpha}^l$ for fixed l and i is also finite. Together with the following claim, this implies that $=_i^l$ has finite index.

Claim. For all $l \geq 0$ and left half-messages α, α' with $\alpha \neq \varepsilon$, $\alpha' \neq \varepsilon$, and $l(\alpha) = l(\alpha') = l$: $M_{i,\alpha}^l = M_{i,\alpha'}^l$ implies $\alpha =_i^l \alpha'$.

Proof of the claim. Assume $M_{i,\alpha}^l = M_{i,\alpha'}^l$. 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 $\in_h$, their exist right half-messages γ, 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 23 implies $l' := l(\beta) \leq \mathsf{depth}(\mathcal{A}_i) \cdot (l+1)$.

Now, using $M^l_{i,\alpha} = M^l_{i,\alpha'}$, it follows $[(\alpha,\beta)]_{=_i^{l,l'}} \in M^{l,l'}_{i,\alpha} = M^{l,l'}_{i,\alpha'}$. Thus, there exists a left half-message β' with $l(\beta') = l'$ such that $[(\alpha,\beta)]_{=_i^{l,l'}} = [(\alpha',\beta')]_{=_i^{l,l'}}$. In particular, $\beta \equiv^{l'}_{i+1} \beta'$, and from $p(\alpha,\beta)q \in_h \mathcal{A}_i$ it follows $p(\alpha',\beta')q \in_h \mathcal{A}_i$. This concludes the proof of the claim.

It is easy to see that $I(\equiv^r_i) \leq I(\equiv^{r'}_i)$ and $I(=^r_i) \leq I(=^{r'}_i)$, for every $r \leq r'$. From this we can conclude

$$\begin{aligned} I(=^l_i) &\in \mathcal{O}\left(I\left(=_i^{l,\mathsf{depth}(\mathcal{A}_i)\cdot(l+1)}\right)^{\mathsf{depth}(\mathcal{A}_i)\cdot(l+1)}\right) \\ &\subseteq \mathcal{O}\left(\left(I\left(\equiv^{\mathsf{depth}(\mathcal{A}_i)\cdot(l+1)}_{i+1}\right)\cdot 2^{|Q_i|^2}\right)^{\mathsf{depth}(\mathcal{A}_i)\cdot(l+1)}\right) \end{aligned}$$

Now it is easy to show that the index of $\equiv^l_i$ is finite. Consider the mapping f^l_i that takes a left half-message α of level l to the tuple

$$([\alpha]_{\equiv^l_{i+1}}, [\alpha^*]_{\equiv_i}, [\alpha_1]_{=^1_i}, \ldots, [\alpha_l]_{=^l_i}, [x]_{=_i}),$$

where x is a message such that $\alpha = x\alpha_l$. Obviously, the range of f^l_i is finite. Finally, it is straightforward to prove for left half-messages α, α' with $l(\alpha) = l(\alpha') = l$ that $f^l_i(\alpha) = f^l_i(\alpha')$ implies $\alpha \equiv^l_i \alpha'$. From this, the bound on $I(\equiv^l_i)$ claimed in the proposition follows immediately. ∎

7.5 Proof of Theorem 11

Putting everything together, we now show that the path problem is decidable. This will conclude the proof of Theorem 11.

More precisely, 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 $(h, \mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$ has the solution $(m_i, m'_i, \ldots, m_{k-1}, m'_{k-1})$, and then construct a solution with short paths (and messages). 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

$$\pi = q_0(a_0,b_0)q_1(a_1,b_1)\cdots(a_{r-1},b_{r-1})q_r \tag{4}$$

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 $q_r = q^F_i$. Let $m_i(j,j') := a_j\cdots a_{j'-1}$ 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. Section 7.3).

We 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 18, let n be the number of times a message in M was used to derive $m_i \in \mathsf{d}_h(\mathcal{K})$ 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. Also, let $v_0,\ldots,v_{n-1}$ be pairwise distinct variables. Then, there exists a term $t \in \mathsf{d}_h(N \cup \{v_0,\ldots,v_{n-1}\})$, such that every variable v_j, $j < n$, occurs

exactly once in t and $m_i = t[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$. Moreover, there exist positions $i_j, i'_j \leq r$ 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, x_j is being read between the positions i_j and i'_j in π.

Finally, we define a mapping f_π as follows: For every $j \leq r$,

$$f_\pi(j) := \begin{cases} (q_j, l_j, l'_j, [m'_i(0,j)]_{\equiv^{l'_j}_{i+1}}, x_s, m_i(i_s, j)), & \text{there exists } s < n \text{ s.t. } i_s < j < i'_s; \\ (q_j, l_j, l'_j, [m'_i(0,j)]_{\equiv^{l'_j}_{i+1}}), & \text{otherwise} \end{cases}$$

This mapping indicates, at which positions π can be cut through. It distinguishes between positions "inside" and "outside" an x_j. The following lemma makes this precise.

Lemma 31 *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 $(h, \mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$ such that the path $\overline{\pi}$:=*

$$q_0(a_0, b_0) q_1 \cdots q_{j_u}(a_{j_{u+1}}, b_{j_{u+1}}) q_{j_{u+1}+1}(a_{j_{u+1}+1}, b_{j_{u+1}+1}) \cdots (a_{r-1}, b_{r-1}) q_r$$

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 25 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 9) it follows that $\overline{m}'_i := m'_i(0, j_u) m'_0(j_{u+1}, r)$ must be a message.

Obviously, $\overline{\pi}$ is a path from q_i^I to q_i^F with input label $\overline{m}_i$ and output label $\overline{m}'_i$.

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 $(h, \mathcal{K}, \mathcal{A}_i, \ldots, A_{k-1})$.

We first show $\overline{m}_i \in \mathsf{d}_h(\mathcal{K})$: 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. In this way, the two extra components in the first tuple of the definition of f_π prohibit that only part of an x_s is removed when going from π to π'. From this, it is easy to conclude that there exists a term $\overline{t} \in \mathsf{d}_h(N \cup \{v_0, \ldots, v_{n-1}\})$, in which every variable v_j, $j < n$, occurs at most once, such that $\overline{m}_i := \overline{t}[v_0/x_0, \ldots, v_{n-1}/x_{n-1}]$. Thus, $\overline{m}_i \in \mathsf{d}_h(\mathcal{K})$.

Because $f_\pi(j_u) = f_\pi(j_{u+1})$, we know $[m'_i(0, j_u)]_{\equiv^{l'}_{i+1}} = [m'_i(0, j_{u+1})]_{\equiv^{l'}_{i+1}}$ and $l' := l'_{j_u} = l'_{j_{u+1}}$. With Proposition 22 we can conclude

$$m'_i = m'_i(0, j_{u+1}) m'_i(j_{u+1}, r) \equiv_{i+1} m'_i(0, j_u) m'_i(j_{u+1}, r) = \overline{m}'_i.$$

Now, Proposition 18 guarantees that the instance $(h, \overline{\mathcal{K}}, \mathcal{A}_{i+1}, \ldots, A_{k-1})$ with $\overline{\mathcal{K}} := \mathcal{K} \cup \{\overline{m}'_i\}$ has a solution $(\overline{m}_{i+1}, \overline{m}'_{i+1}, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1})$, and thus, $(\overline{m}_i, \overline{m}'_i, \overline{m}_{i+1}, \overline{m}'_{i+1}, \ldots, \overline{m}_{k-1}, \overline{m}'_{k-1})$ is a solution of $(h, \mathcal{K}, \mathcal{A}_i, \ldots, \mathcal{A}_{k-1})$. ∎

We now need to show that the range of f_π is finite. We know that the depth $\mathsf{depth}_h(\mathcal{K})$ of the messages in $\mathsf{d}_h(\mathcal{K})$ is bounded by the size of $\mathcal{K}$ and h (Observation 2). In particular, $l_\pi(j) \leq \mathsf{depth}_h(\mathcal{K})$ for every $j \leq r$ and π as defined in (4). Then, Corollary 24 implies that the depth of the messages returned by $\mathcal{A}_i$ is bounded by $\mathsf{depth}(\mathcal{A}_i) \cdot (\mathsf{depth}_h(\mathcal{K}) + 1)$. This means, $l'_\pi(j) \leq \mathsf{depth}(\mathcal{A}_i) \cdot (\mathsf{depth}_h(\mathcal{K}) + 1)$ for every $j \leq r$. Now with Proposition 30, it is easy to see that the range of f_π is finite, i.e., bounded in the size of the problem instance. Consequently, with Lemma 31 the length of the paths $q_i^I(m_i, m'_i)q_i^F \in \mathcal{A}_i$ can be bounded as well (and this bound can be computed effectively). Clearly, for an instance $(h, \mathcal{K}, \mathcal{A}_0, \ldots, \mathcal{A}_{k-1})$ of PathProblem this holds for every $i < k$, and thus, we conclude:

Corollary 32 PathProblem *is decidable.*

As an immediate consequence, we obtain Theorem 11.

8 Conclusion

An important feature of cryptographic group protocols is that principals need to process messages with an (a priori) unbounded number of data fields. Most protocol models proposed in the literature, in particular those in which security is decidable, are, however, restricted to principals which can only process input messages with a fixed and finite format. To overcome this problem we have introduced a generic model for cryptographic group protocols, in which protocols can perform arbitrary (nondeterministic) actions and we have investigated decidability for different instances of this generic model. It turned out that if actions of principals are modeled by sets of rewrite rules, which can nondeterministically be applied by a principal to the input messages, security is undecidable. On the other hand, and this is the main technical result of this paper, if principals are modeled by so-called message-transducers, the problem becomes decidable, provided that the nesting depth of encryptions and hashes performed by an intruder is restricted. These results have been shown for the shared key setting and secrecy properties; we conjecture that they carry over rather easily to public key encryption and authentication.

There are different directions for future work. A very interesting question is, whether in the transducer-based model we obtain decidability even for an unrestricted intruder. Also, it is open how complex keys can be handled in our setting. From a practical point of view, it remains to investigate whether our algorithm for deciding the existence of h-attacks on transducer-based protocols can serve as a basis for a practical decision procedure.

Acknowledgement I thank Thomas Wilke for many helpful comments on this work, and Cathrine Meadows for pointing me to the paper by Pereira and Quisquater.

References

[1] R.M. Amadio, D. Lugiez, and V. Vanackère. On the symbolic reduction of processes with cryptographic functions. Technical Report RR-4147, INRIA, 2001.

[2] G. Ateniese, M. Steiner, and G. Tsudik. Authenticated group key agreement and friends. In *Proceedings of the 5th ACM Conference on Computer and Communication Secruity (CCS'98)*, pages 17–26, San Francisco, CA, 1998. ACM Press.

[3] J. Bryans and S.A. Schneider. CSP, PVS, and a Recursive Authentication Protocol. In *DIMACS Workshop on Formal Verification of Security Protocols*, 1997.

[4] J.A. Bull and D.J. Otway. The authentication protocol. Technical Report DRA/CIS3/PROJ/CORBA/SC/1/CSM/436-04/03, Defence Research Agency, Malvern, UK, 1997.

[5] I. Cervesato, N.A. Durgin, P.D. Lincoln, J.C. Mitchell, and A. Scedrov. A Meta-notation for Protocol Analysis. In *12th IEEE Computer Security Foundations Workshop (CSFW12)*, 1999.

[6] D. Dolev, S. Even, and R.M. Karp. On the Security of Ping-Pong Protocols. *Information and Control*, 55:57–68, 1982.

[7] N.A. Durgin, P.D. Lincoln, J.C. Mitchell, and A. Scedrov. Undecidability of bounded security protocols. In *Workshop on Formal Methods and Security Protocols (FMSP'99)*, 1999.

[8] S. Even and O. Goldreich. On the Security of Multi-Party Ping-Pong Protocols. Technical Report 285, Technion – Israel Institut of Technology, 1983.

[9] N. Ferguson and B. Schneier. A Cryptographic Evaluation of IPsec. Technical report, 2000. To appear.

[10] A. Huima. Efficient infinite-state analysis of security protocols. In *Workshop on Formal Methods and Security Protocols (FMSP'99)*, 1999.

[11] R. Küsters. On the Decidability of Cryptographic Group Protocols. Technical Report 0201, Institut für Informatik und Praktische Mathematik, CAU Kiel, Germany, 2002.
Available from http://www.informatik.uni-kiel.de/reports/2002/0201.html.

[12] C. Meadows. Extending formal cryptographic protocol analysis techniques for group protocols and low-level cryptographic primitives. In P. Degano, editor, *Proceedings o the First Workshop on Issues in the Theory of Security (WITS'00)*, pages 87–92, 2000.

[13] C. Meadows. Open issues in formal methods for cryptographic protocol analysis. In *Proceedings of DISCEX 2000*, pages 237–250. IEEE Computer Society Press, 2000.

[14] D. Otway and O. Rees. Efficient and timely mutual authentication. *Operating Systems Review*, 21(1):8–10, January 1987.

[15] L.C. Pauslon. Mechanized proofs for a recursive authetication protocol. In *10th IEEE Computer Security Foundations Workshop (CSFW10)*, pages 84–95, 1997.

[16] O. Pereira and J.-J. Quisquater. A Security Analysis of the Cliques Protocols Suites. In *Proceedings of the 14th IEEE Computer Security Foundations Workshop (CSFW'14)*, pages 73–81, 2001.

[17] M. Rusinowitch and M. Turuani. Protocol Insecurity with Finite Number of Sessions is NP-complete. In *14th IEEE Computer Security Foundations Workshop (CSFW15)*, 2001.

[18] M. Steiner, G. Tsudik, and M. Waidner. CLIQUES: A new approach to key agreement. In *IEEE International Conference on Distributed Computing Systems*. IEEE Computer Society Press, 1998.

[19] J. Zhou. Fixing a security flaw in IKE protocols. *Electronic Letter*, 35(13):1072–1073, 1999.