Available online at www.sciencedirect.com
SCIENCE DIRECT®
Theoretical Computer Science 338 (2005) 393–425
Theoretical Computer Science
www.elsevier.com/locate/tcs

A method for symbolic analysis of security protocols☆

Michele Boreale[a], Maria Grazia Buscemi[b,*]

[a]*Dipartimento di Sistemi e Informatica, Università di Firenze, Italy*
[b]*Dipartimento di Informatica, Università di Pisa, Via Filippo Buonarroti, 2, 56127 Pisa, Italy*

Received 10 May 2004; received in revised form 31 January 2005; accepted 3 March 2005

Communicated by P.S. Thiagarajan

Abstract

In security protocols, message exchange between the intruder and honest participants induces a form of state explosion which makes protocol models infinite. We propose a general method for automatic analysis of security protocols based on the notion of *frame*, essentially a rewrite system plus a set of distinguished terms called *messages*. Frames are intended to model generic crypto-systems. Based on frames, we introduce a process language akin to Abadi and Fournet's applied pi. For this language, we define a symbolic operational semantics that relies on unification and provides finite and effective protocol models. Next, we give a method to carry out trace analysis directly on the symbolic model. We spell out a *regularity* condition on the underlying frame, which guarantees completeness of our method for the considered class of properties, including secrecy and various forms of authentication. We show how to instantiate our method to some of the most common crypto-systems, including shared- and public-key encryption, hashing and Diffie–Hellman key exchange.

Keywords: Security protocol analysis; Symbolic techniques; Process calculi

☆ Extended and revised version of [8–10]. This work has been partially supported by EU within the IST FET—Global Computing initiative, Projects MIKADO and PROFUNDIS.

* Corresponding author.
E-mail addresses: boreale@dsi.unifi.it (M. Boreale), buscemi@di.unipi.it (M.G. Buscemi).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2005.03.044

1. Introduction

Many of the methods employed in security protocol analysis are based conceptually on a model dating back to Dolev and Yao [18], where a (hostile) intruder has total control over the communication network. In particular, it is assumed that the intruder can learn, hide or replace any message in transit on the network. It can also synthesize new messages starting from learned messages and using arbitrary combinations of operations like nonce creation, pairing, encryption and decryption. The intruder cannot guess secret keys or forge messages it cannot synthesize. Thus, sending a message on the network means handing it to the intruder, while receiving a message from the network means accepting any message the intruder can synthesize at a given moment. Due to the latter point, any Dolev–Yao model is in principle infinite.

Traditional finite-state model checking has been employed in security protocol analysis (e.g., [26,32,36,39]), under two simplifying assumptions: (a) there is a bound on the number of protocol runs, and (b) at any moment, there is a bound on the number of possible messages the intruder can synthesize and send to honest participants. Discarding either of these two assumptions leads to infinite models. Also, these bounds have to be chosen carefully: due to the combinatorics of message generation, the size of the model tends to explode as the number of principals and data values increases.

In general, it is known that discarding assumption (a) leads to undecidability of protocol analysis, unless severe syntactic restrictions are imposed on the analysed protocols (see e.g. [4,15,19,20,23,34]). In particular, in the presence of pairing and encryption, an even weak form of iteration (the ability to create arbitrarily many protocol instances) allows for encoding of 2-counter machines, which in turn implies undecidability of e.g. secrecy, based on information transfer from one protocol instance to another (*blind copying*) [35]. Wanting to preserve decidability *and* an expressive term language, one is left with little choice but keeping assumption (a), hence ruling iteration out.

In the last few years, symbolic approaches have been proposed that make infinite-state analysis possible and lead to discard assumption (b) [4,8,15,30]. These approaches focus on specific crypto-systems (typically, shared- or public-key encryption), and the corresponding completeness proofs are rather ad hoc. The present paper introduces a general framework for symbolic protocol analysis. It can be viewed as an attempt at presenting in a uniform manner methods based on unification (e.g. [8,4]), while extracting a common factor out of the related proof techniques. When instantiated to specific crypto-primitives, under a condition of regularity that we illustrate below, the framework yields complete verification methods. In those case studies that we have actually experimented [9], the method is also quite effective in practice.

More in detail, we start by introducing a notion of *frame*, essentially a term rewriting system plus a set of distinguished terms called *messages*. We consider a generic signature Σ that may include constructors and destructors for various cryptographic operations. The meaning of Σ-terms is provided by an evaluation relation $\downarrow$ that maps terms to messages. On top of the evaluation relation, we introduce a deduction relation $\vdash$ that describes how the environment can synthesize new messages from known ones. On top of a generic frame, we introduce a process language akin to Abadi and Fournet's applied pi [1], that can be used to describe protocols. Protocol properties are formalised as correspondence assertions

between I/O events, or *actions*, of the form "every execution of action α must be preceded by some execution of action β".

In agreement with the Dolev–Yao approach, the "concrete" operational semantics of the process calculus is infinitary, because each input action gives rise to infinitely many transitions. This problem is overcome by introducing a symbolic operational semantics. In the latter, as a result of a receive operation, input variables are not instantiated, rather they are *constrained* as the computation proceeds. Constraints are generated by symbolic evaluation of terms representing crypto-operations, and take the form of most general unifiers (mgu's) between terms of the signature. As an example, evaluation of shared-key decryption of ζ using key η, written $\mathsf{dec}_\eta(\zeta)$, generates a mgu θ for the equation $\zeta = \{x\}_\eta$, for a fresh x. The result of the decryption is therefore represented as $\theta(x)$. Mgu's are propagated through whole process terms as soon as they are generated. The resulting transition system is finitely branching, hence it yields finite models when protocols with a finite number of participants are considered.

Next, we give a method to carry out trace analysis directly on the symbolic model, and provide a *regularity* condition on the given frame, under which the method is proven sound and complete with respect to the concrete semantics. In other words, for regular frames every attack detected in the symbolic model corresponds to some attack in the concrete one, and vice-versa. Thus, our method makes no approximation with respect to the infinitary, concrete model. For instance, type-dependent flaws (see e.g. [22]), which usually escape finite-state analysis, with our approach naturally emerge when present. The regularity condition roughly amounts to requiring that the set of messages deducible from any trace of the protocol can be *syntactically* built out of a finite basis of messages, and that the induced finite-basis operation commutes with substitution.

We show how to instantiate the general framework to some of the most common cryptosystems, providing frames for shared- and public-key encryption, digital signature, hashing and Diffie–Hellman exponentiation. The proof of regularity is covered in detail for the public-key frame only. We also highlight the relevance of the regularity condition by providing an example of a meaningful non-regular frame. This also illustrates the limits of our approach.

Our method is quite efficient in practice, because in the symbolic model there is no state-explosion induced by message exchange: every input action gives rise exactly to one symbolic transition. We have developed a prototype tool, STA (Symbolic Trace Analyzer), based on this method [41]. Experimentation with STA has given very encouraging results [9].

Related work: Early work on symbolic analysis is due to Huima. In [24], the execution of a protocol generates a set of equational constraints. Only an informal description is provided of the kind of equational rewriting needed to solve these constraints. Approaches based on symbolic analysis were also exploited in [8,3,21], all of which focus on shared-key encryption. The work [8] introduces a shared-key only version of our symbolic method. In [3], unlike our approach, symbolic execution and consistency check are not kept separate, and this may have a relevant impact on the size of the computed symbolic model. Another point worth noting is that, in [3], a brute-force method is needed to resolve variables in key position: such variables have to be instantiated to every possible name used by the participants; this fact may lead to state explosion, too. In [21], a procedure is provided to analyse the

knowledge of the environment, based on a symbolic semantics akin to [8]. The approach applies to protocols with arbitrary messages as keys, but, like ours, it is proven complete only for atomic keys. Also, the method suffers from the same problem as [3] concerning brute-force instantiation. The paper [4] extends the symbolic reachability analysis of [3] to hash functions and public key cryptography and establishes some complexity results.

Developments of the symbolic approach not specifically relying on unification are presented in [15,30]. The decision technique in [15] is based on a reduction to a set constraint problem which is in turn reduced to an automata-theoretic problem. Completeness is proven by assuming rather severe restrictions on protocol syntax. The technique in [30] focuses on reachability properties and is based on constraint solving; the approach makes use of the strand space formalism [42] to specify protocol processes. The symbolic reduction and the knowledge analysis are separated and the latter is performed by a procedure for constraint solving procedure.

Some recent papers [7,16,33,34] focus on protocols with unbounded instances and unbounded message size, and give verification algorithms that terminate under certain assumptions, like tagging. Other recent work addresses the symbolic analysis problem in the presence of low-level cryptographic operations and, in particular, modular exponentiation [31,13,40]. Blanchet's model [6] abstracts away from operations like inverse, root extraction and random number generation. The resulting method may give rise to false attacks and may not terminate. Pereira and Quisquater first [31] proposed a technique for analysing group Diffie–Hellman protocols in the presence of an attacker with restricted capabilities (e.g. no symmetric encryption), though not facing the issue of decidability. Chevalier et al. [13] demonstrated that the protocol analysis problem is decidable and NP-complete in the presence of modular exponentiation. Shmatikov [40] proved that the above problem in the presence of Abelian group operator and exponentiation is decidable for a finite number of protocol sessions. Also related to these approaches is protocol analysis in the presence of the xor operation, which has been recently proven to be decidable by Chevalier et al. [12] and, independently, by Comon-Lundh and Shmatikov [17].

Summary: In Section 2 we introduce the notion of frame at the basis of our method. In Section 3 we present the process language, its concrete and symbolic semantics, and we study the relationship between the two semantics. In Section 4 we describe the verification method based on the symbolic semantics. Throughout Sections 2–4 we use public-key cryptography as a running example. An extended system featuring shared-key, public-key, digital signature and hashing is considered in Section 5; this section also contains an example of non-regular frame. In Section 6 we illustrate an application to a low-level primitive, modular exponentiation, hence, the Diffie–Hellman key exchange. Section 7 illustrates STA on the classic Needham–Schroeder protocol. In Section 8 we draw some conclusions. Detailed proofs of a few technical results are confined to Appendices A, B and C.

2. A general framework

In this section, we present the main ingredients of our framework. We introduce the concept of *frame*, that is, a structure consisting of a signature, a set of (legal) messages

and an evaluation relation. Then, we define the notions of process, trace, configuration and security property.

2.1. Frames

We consider two countable disjoint sets of *names* $m, n, \ldots \in \mathcal{N}$ and *variables* $x, y, \ldots \in \mathcal{V}$. The set $\mathcal{N}$ is in turn partitioned into a countable set of *local names* $a, b, \ldots \in \mathcal{LN}$ and a countable set of *environmental names* $\underline{a}, \underline{b}, \ldots \in \mathcal{EN}$: these two sets represent infinite supplies of fresh quantities (keys, nonces, ...), that can be used by processes and environment, respectively. The set $\mathcal{N} \cup \mathcal{V}$ is ranged over by letters $u, v, \ldots$. The fact that $\mathcal{LN}$ and $\mathcal{EN}$ are disjoint guarantees that nonces and keys generated by honest participants cannot be guessed in advance by the environment (of course, local names might be learned and then used by the environment), and vice-versa.

Given a finite signature Σ of function symbols $f, g, \ldots$, each coming with its arity (constants have arity 0), we denote by $\mathcal{E}_\Sigma$ the algebra of terms (or *expressions*) on $\mathcal{N} \cup \mathcal{V} \cup \Sigma$, given by the grammar:

$$\zeta, \eta \quad ::= \quad u \mid f(\widetilde{\zeta}),$$

where $\widetilde{\zeta}$ is a tuple of terms of the expected length. A *term context* $C[\cdot]$ is a term with a hole that can be filled with any term ζ, thus yielding a term $C[\zeta]$.

Definition 1 (*Frame*). A *frame* $\mathcal{F}$ is a triple $(\Sigma, \mathcal{M}, \downarrow)$, where:
- Σ is a signature;
- $\mathcal{M} \subseteq \mathcal{E}_\Sigma$ is a set of *messages* $M, N, \ldots$;
- $\downarrow \subseteq \mathcal{E}_\Sigma \times \mathcal{E}_\Sigma$ is an *evaluation relation*.

In the sequel, we write $\zeta \downarrow \eta$ for $(\zeta, \eta) \in \downarrow$ and say that ζ *evaluates to* η. In typical frame instances the relation $\downarrow$ will be both a function and a congruence with respect to the operations in Σ, but we need not to assume these facts in the general framework. In fact, as we shall see in Section 6, a non-deterministic evaluation relation can be used to model a commutative operation.

Next, we define a deduction relation ($\vdash$), which specifies how the environment can generate new messages starting from an initial set of messages S. Our definition of deduction relation is not given by a set of deductive rules. Rather, we make use of the set $\mathcal{H}(S)$, which consists of all the expressions inductively built by applying functions of Σ to elements of S and of $\mathcal{EN}$. We denote by $\mathcal{P}_f(X)$ the set of finite subsets of X.

Definition 2 (*Deduction relation*). For $\mathcal{F} = (\Sigma, \mathcal{M}, \downarrow)$ a frame and $S \subseteq \mathcal{M}$, the set $\mathcal{H}_\mathcal{F}(S)$ is inductively defined by the following clauses:

$$\begin{aligned}
\mathcal{H}^0_\mathcal{F}(S) &= S \cup \mathcal{EN} \\
\mathcal{H}^{i+1}_\mathcal{F}(S) &= \mathcal{H}^i_\mathcal{F}(S) \cup \{ f(\widetilde{\zeta}) : \ f \in \Sigma, \ \widetilde{\zeta} \subseteq \mathcal{H}^i_\mathcal{F}(S) \} \\
\mathcal{H}_\mathcal{F}(S) &= \bigcup_{i \geq 0} \mathcal{H}^i_\mathcal{F}(S).
\end{aligned}$$

Table 1
$\mathcal{F}_{pk}$, a frame for public key encryption

$$\text{SIGNATURE} \quad \Sigma = \{(\cdot)^+, \quad (\cdot)^-, \quad \{\!|\cdot|\!\}_{(\cdot)}, \quad \langle\cdot,\cdot\rangle, \quad \pi_i(\cdot) \quad (i=1,2), \quad \mathsf{dec}^{\mathsf{pk}}_{(\cdot)}(\cdot)\}$$

$$\text{MESSAGES} \quad M, N ::= u \mid u^+ \mid u^- \mid \{\!|M|\!\}_{u^+} \mid \langle M, N\rangle$$

$$\text{(PRJ)} \quad \pi_i(\langle\zeta_1,\zeta_2\rangle) \rightsquigarrow \zeta_i \quad (i=1,2)$$

$$\text{(DEC)} \quad \mathsf{dec}^{\mathsf{pk}}_{\eta^-}(\{\!|\zeta|\!\}_{\eta^+}) \rightsquigarrow \zeta$$

$$\text{(CTX)} \quad \frac{\zeta \rightsquigarrow \zeta'}{C[\zeta] \rightsquigarrow C[\zeta']}$$

$$\text{EVALUATION} \quad \zeta \downarrow \eta \quad \triangleq \quad \zeta \rightsquigarrow^* \eta$$

The *deduction relation* $\vdash_{\mathcal{F}} \subseteq \mathcal{P}_f(\mathcal{M}) \times \mathcal{M}$ is defined by

$$S \vdash_{\mathcal{F}} M \quad \triangleq \quad \exists\, \zeta \in \mathcal{H}_{\mathcal{F}}(S) \,:\, \zeta \downarrow M.$$

A message M is *deducible* from S if $S \vdash_{\mathcal{F}} M$.

When no confusion arises, we simply write $\mathcal{H}(S)$ for $\mathcal{H}_{\mathcal{F}}(S)$ and $\vdash$ for $\vdash_{\mathcal{F}}$.

Example 1 (*Public-key encryption*). A frame $\mathcal{F}_{pk} = (\Sigma, \mathcal{M}, \downarrow)$ for public-key cryptography is defined in Table 1. The functions of Σ are: generation of public ($(\cdot)^+$) and private ($(\cdot)^-$) keys, encryption with a public key ($\{\!|\cdot|\!\}_{(\cdot)}$), decryption using a private key ($\mathsf{dec}^{\mathsf{pk}}_{(\cdot)}(\cdot)$), pairing ($\langle\cdot,\cdot\rangle$) and selection ($\pi_i(\cdot)$). Public and private keys are represented by u^+ and u^-, respectively. Names and variables can be used to build compound messages via public-key encryption and pairing. In particular, $\{\!|M|\!\}_{m^+}$ represents the message obtained by encrypting M under m^+. Primitives for pairing and public key encryption of messages can be arbitrarily nested. Non-atomic keys are forbidden in messages: this restriction is crucial in our method, as we will show in Example 7. The definition of evaluation relation makes use of an auxiliary relation $\rightsquigarrow$, that models the mechanisms of public key encryption under the perfect cryptography assumption (see e.g. [18]).

As an example of deduction, if $S = \{\, \{\!|\langle a, b\rangle|\!\}_{k^+}, k^- \,\}$ then $S \vdash a$, since $\zeta = \pi_1(\mathsf{dec}^{\mathsf{pk}}_k - (\{\!|\langle a, b\rangle|\!\}_{k^+})) \in \mathcal{H}(S)$ and $\zeta \downarrow a$. Note that, whatever S, the set of messages deducible from S is infinite.

Table 2
Syntax for agents

AGENTS $\mathcal{A}$	A, B	::=	0	(null)
		|	$\mathsf{a}(x).\, A$	(input)
		|	$\overline{\mathsf{a}}\langle\zeta\rangle.\, A$	(output)
		|	$\mathsf{let}\ y{=}\zeta\mathsf{in}\ A$	(evaluation)
		|	$[\zeta = \eta]A$	(matching)
		|	$A \parallel B$	(parallel composition)

The occurrences of variables x and y are bound.

2.2. Processes

2.2.1. Syntax

As a base language, we consider a variant of the applied pi-calculus [1], parametrised by an arbitrary frame $\mathcal{F}$ (for readability, we omit explicit reference to $\mathcal{F}$ in the notation). The syntax of *agent expressions*, whose set we name $\mathcal{A}$, is reported in Table 2. A single construct (let) for expression evaluation replaces the ad hoc constructs found in the spi-calculus for encryption, decryption and other cryptographic operations. The main difference from applied pi is that, here, we consider a set $\mathcal{L}$ of input and output *labels*, ranged over by $\mathsf{a}, \mathsf{b}, \ldots$, which must not be regarded as channels—according to the Dolev–Yao model, we assume just one public network—but, rather, as 'tags' attached to process actions for ease of reference. We do not consider the pi-calculus restriction operator: it could be easily accommodated, but it has no semantic relevance, in the absence of iteration.

Given the presence of binders for variables, notions of *free variables*, $\mathrm{v}(A) \subseteq \mathcal{V}$, and *alpha-equivalence* arise as expected. We shall identify alpha-equivalent agent expressions. For any ζ and x, $[\zeta/x]$ denotes the operation of substituting the free occurrences of x by ζ. An agent expression A is said to be *closed* or a *process* if $\mathrm{v}(A) = \emptyset$; the set of processes $\mathcal{P}$ is ranged over by $P, Q, \ldots$. Local names and environmental names occurring in A are denoted by $\mathrm{ln}(A)$ and $\mathrm{en}(A)$, respectively. A process P is *initial* if $\mathrm{en}(P) = \emptyset$.

Example 2 (*The Needham–Schroeder protocol*). We consider the classical Needham–Schroeder protocol as described, e.g. in [26]. The protocol involves two honest participants, A and B, which want to authenticate with one another. A is the initiator, B the responder:
(1) $A \longrightarrow B : \{\!|nA, id_A|\!\}_{kB^+}$ (nA fresh nonce),
(2) $B \longrightarrow A : \{\!|nA, nB|\!\}_{kA^+}$ (nB fresh nonce),
(3) $A \longrightarrow B : \{\!|nB|\!\}_{kB^+}$.
We formalise below a 'one-shot' configuration of this protocol, *NS*, where two distinct instances of A are willing to talk to B and to a malicious insider I, a participant whose role is played by the attacker. An instance of B is willing to respond to A (this example will be analysed in Section 7). A disclose action is supposed to have provided the environment with its initial knowledge (identities and public keys of participants, plus the insider's private key

information). To simplify the notation, we use a few self-explaining notational shorthands, like $\mathsf{c}(\{\!|y, n|\!\}_{k^+}).\ P$ for

$$\mathsf{c}(x).\,\mathsf{let}\ x'=\mathsf{dec}^{\mathsf{pk}}_{k^-}(x)\mathsf{in}\,\mathsf{let}\ y=\pi_1(x')\mathsf{in}\,\mathsf{let}\ y'=\pi_2(x')\mathsf{in}\,[y'=n]P,$$
$$\text{for fresh } x, x', y'.$$

$$A \triangleq \overline{\mathsf{a1}}\langle\{\!|nA, \mathtt{id_A}|\!\}_{kB^+}\rangle.\,\mathsf{a2}(\{\!|nA, xnB|\!\}_{kA^+}).\,\overline{\mathsf{a3}}\langle\{\!|xnB|\!\}_{kB^+}\rangle.\,0$$
$$\quad || \quad \overline{\mathsf{a'1}}\langle\{\!|n'A, \mathtt{id_A}|\!\}_{kI^+}\rangle.\,\mathsf{a'2}(\{\!|n'A, xnI|\!\}_{kA^+}).\,\overline{\mathsf{a'3}}\langle\{\!|xnI|\!\}_{kI^+}\rangle.\,0$$

$$B \triangleq \mathsf{b1}(\{\!|ynA, \mathtt{id_A}|\!\}_{kB^+}).\,\overline{\mathsf{b2}}\langle\{\!|ynA, nB|\!\}_{kA^+}\rangle.\,\mathsf{b3}(\{\!|nB|\!\}_{kB^+}).\,0$$

$$NS \triangleq \langle\overline{\mathsf{disclose}}\langle kI, kA^+, kB^+, \mathtt{id_A}, \mathtt{id_B}, \mathtt{id_I}\rangle,\ (A \,||\, B)\rangle\,.$$

2.2.2. Operational semantics

The semantics of the calculus is given in terms of a transition relation $\longrightarrow$, which we will sometimes refer to as 'concrete' (as opposed to the 'symbolic' one we shall introduce later on). We model the state of the system as a pair $\langle s,\ P\rangle$, where s records the current environment's knowledge (i.e., the sequence of messages the environment has "seen" on the network up to a given moment) and P is a process term. An *action* is a term of the form $\mathsf{a}\langle M\rangle$ (*input* action) or $\overline{\mathsf{a}}\langle M\rangle$ (*output* action), for a a label and M a message. The set of actions Act is ranged over by $\alpha, \beta, \ldots$, while the set Act^* of strings of actions is ranged over by $s, s', \ldots$. String concatenation is written '$\cdot$' . We denote by $\mathrm{act}(s)$ and $\mathrm{msg}(s)$ the set of actions and messages, respectively, appearing in s. A string s is *closed* if $\mathrm{v}(s) = \emptyset$ and *initial* if $\mathrm{en}(s) = \emptyset$. In what follows, we shall often write '$s \vdash M$' for $\mathrm{msg}(s) \vdash M$ and '$M \in s$' for $M \in \mathrm{msg}(s)$.

Below we define *traces*, i.e. sequences of actions that may result from the interaction between a process and its environment. In traces, each message received by a process (input message) must be synthesizable from the knowledge the environment has previously acquired. In *configurations*, the environment's knowledge is explicitly recorded as a trace.

Definition 3 (*Traces and configurations*). A *trace* is a closed string $s \in Act^*$ such that for each s_1, s_2 and $\mathsf{a}\langle M\rangle$, if $s = s_1\cdot\mathsf{a}\langle M\rangle\cdot s_2$ then $s_1 \vdash M$.

A *configuration*, written as $\langle s,\ P\rangle$, is a pair consisting of a trace s and a process P. A configuration is *initial* if $\mathrm{en}(s,\ P) = \emptyset$. Configurations are ranged over by $\mathcal{C}, \mathcal{C}', \ldots$.

The concrete transition relation on configurations is defined by the rules in Table 3. Each action taken by a process is recorded in the configuration's first component. Rule (INP) makes the transition relation infinitely-branching, as M ranges over the infinite set $\{M\ :\ s \vdash M,\ M \text{ closed}\}$. In rule (OUT), ζ is evaluated before the action takes place. By rule (LET), the evaluation of ζ replaces any occurrence of y in P. Note that, while we require that evaluation of terms sent on the network yields closed messages, for the purpose of internal computation (rules (LET) and (MATCH)) we do allow evaluation to arbitrary terms. No handshake communication is provided: all messages go through the environment (rule (PAR)). By $\mathcal{C} \longrightarrow^n \mathcal{C}'$ we mean that $\mathcal{C}$ reduces to $\mathcal{C}'$ in n execution steps.

Table 3
Rules for the transition relation ($\longrightarrow$)

(INP)	$\langle s,\ \mathsf{a}(x).P\rangle \longrightarrow \langle s{\cdot}\mathsf{a}\langle M\rangle,\ P[^{M}/_{x}]\rangle$	$s \vdash M,\ M$ closed
(OUT)	$\langle s,\ \overline{\mathsf{a}}\langle\zeta\rangle.P\rangle \longrightarrow \langle s{\cdot}\overline{\mathsf{a}}\langle M\rangle,\ P\rangle$	$\zeta \downarrow M$, M closed
(LET)	$\langle s,\ \mathsf{let}\ y{=}\zeta\mathsf{in}\ P\rangle \longrightarrow \langle s,\ P[^{\eta}/_{y}]\rangle$	$\zeta \downarrow \eta$, η closed
(MATCH)	$\langle s,\ [\zeta=\eta]P\rangle \longrightarrow \langle s,\ P\rangle$	$\zeta \downarrow \xi,\ \eta \downarrow \xi$
(PAR)	$\dfrac{\langle s,\ P\rangle \longrightarrow \langle s',\ P'\rangle}{\langle s,\ P \parallel Q\rangle \longrightarrow \langle s',\ P' \parallel Q\rangle}$	

plus symmetric version of (PAR).

2.3. Properties

We express security properties of a protocol in terms of the traces generated by the protocol. In particular, we focus on correspondence assertions of the kind 'for every generated trace, whenever action β occurs in the trace, then action α must have occurred at some previous point in the trace'. Given a configuration $\langle s,\ P\rangle$ and a trace s', we say that $\langle s,\ P\rangle$ *generates* s', written $\langle s,\ P\rangle \searrow s'$, if $\langle s,\ P\rangle \longrightarrow^{*} \langle s',\ P'\rangle$ for some P'.

We let ρ range over ground substitutions, i.e. substitutions that map variables to closed messages, and denote by $t\rho$ the result of applying ρ to an arbitrary term t.

Definition 4 (*Satisfaction relation*). Let α and β be actions and s be a trace. We say that α occurs prior to β in s if whenever $s = s'{\cdot}\beta{\cdot}s''$ then $\alpha \in \mathrm{act}(s')$. For $\mathrm{v}(\alpha) \subseteq \mathrm{v}(\beta)$, we write $s \vDash \alpha \hookleftarrow \beta$, and say s *satisfies* $\alpha \hookleftarrow \beta$, if for each ground substitution ρ it holds that $\alpha\rho$ occurs prior to $\beta\rho$ in s. We say that a configuration $\mathcal{C}$ *satisfies* $\alpha \hookleftarrow \beta$, and write $\mathcal{C} \vDash \alpha \hookleftarrow \beta$, if all traces generated by $\mathcal{C}$ satisfy $\alpha \hookleftarrow \beta$.

Assertions $\alpha \hookleftarrow \beta$ can express interesting secrecy and authentication properties. As an example, in the final step of many protocols, a principal A sends a message of the form $\{N\}_k$ to a responder B, where $\{N\}_k$ is obtained by encrypting some authentication information N under a newly established shared-key k. Our scheme permits expressing that *every* message encrypted with k that is accepted by B during the execution of the protocol indeed originates from A, i.e. that B is really talking to A, and that k is authentic. If we denote by final_A and final_B the labels attached to A's and B's final action, respectively, then the above property might be formalised as an assertion $\overline{\mathsf{final}_A}\langle\{x\}_k\rangle \hookleftarrow \mathsf{final}_B\langle\{x\}_k\rangle$, for x a variable.

Example 3 (*Needham–Schroeder Protocol—Cont.*). Consider the protocol configuration *NS* defined in Example 2.The property that, at step 3, B should only accept authentic messages, i.e. messages truly originating from A, is expressed by the following assertion:

$$\mathit{AuthAtoB} \quad \stackrel{\triangle}{=} \quad \overline{\mathsf{a3}}\langle\{\!|z|\!\}_{kB^{+}}\rangle \hookleftarrow \mathsf{b3}\langle\{\!|z|\!\}_{kB^{+}}\rangle,$$

with z fresh in *NS*. This means that any message received by B at step 3 *and* having the form $\{\!|N|\!\}_{kB^+}$, for some N, must have been previously sent by A at step 3. As we shall see in Section 7, property *AuthAtoB* is not satisfied by *NS*.

In practice, all forms of authentication in Lowe's hierarchy [27] are captured by this scheme, except for the most demanding one requiring a one-to-one bijection between α's and β's. However, our scheme can be easily adjusted to include this stronger form, by requiring that each β is preceded by *exactly* one occurrence of α.

Another property that can be set within our framework is *secrecy* in the style of [5]. In this case, it is convenient to fix a conventional 'absurd' action $\perp$ that is nowhere used in agent expressions. Thus, the formula $\perp \leftarrow\!\!\!\circ\ \alpha$ means that action α should never take place. Now, the fact that a protocol, say P, does not leak a sensible datum, say d, can be expressed also by saying that the adversary will never be capable of synthesizing d. This can be formalised by extending the protocol to include a 'guardian' that at any time picks up one message from the network, $P \parallel \mathrm{g}(x).\,0$, and then requiring that this guardian will never receive d, that is, $\langle \varepsilon,\ \ P \parallel \mathrm{g}(x).\,0\rangle \models \perp \leftarrow\!\!\!\circ\ \mathrm{g}\langle d\rangle$. Note that in our framework it is also possible to verify a more general form of secrecy, in which a datum d cannot be leaked until a certain event, represented by a certain action *event*, occurs. This property can be specified by replacing the absurd action above with the *event* action: *event* $\leftarrow\!\!\!\circ\ \mathrm{g}\langle d\rangle$.

3. Symbolic semantics

The symbolic semantics we present in this section is based on the notion of symbolic frame. The latter is essentially a frame equipped with an additional symbolic evaluation relation, which is in agreement with its concrete counterpart.

A substitution θ in a frame $\mathcal{F}$ is a finite partial map from $\mathcal{V}$ to the set of messages $\mathcal{M}$ of frame $\mathcal{F}$ such that $\theta(x) \neq x$, for each variable x. Let us denote by Subst the set of all substitutions in a given frame. For any object t (i.e. variable, message, process, trace,...), we denote by $t\theta$ the result of simultaneously replacing each $x \in \mathrm{v}(t) \cap \mathrm{dom}(\theta)$ by $\theta(x)$. For θ a substitution, we denote by $\mathrm{dom}(\theta)$ and $\mathrm{cod}(\theta)$, the domain and the co-domain of θ. By $\theta_{|V}$, we denote θ restricted to V, i.e. $\{(x, \theta(x)) \mid x \in V\}$. A substitution θ is a *unifier* of t_1 and t_2 if $t_1\theta = t_2\theta$. We denote by $\mathrm{mgu}(t_1, t_2)$ a chosen *most general unifier* (mgu) of t_1 and t_2, that is, a unifier θ of t_1 and t_2 such that any other unifier is a composition of substitution θ with some θ', written $\theta\theta'$.[1] Also, for t_1, t_1', t_2, t_2' terms, $\mathrm{mgu}(t_1 = t_1', t_2 = t_2')$ stands for $(\theta\,\mathrm{mgu}(t_2\theta, t_2'\theta))$, where $\theta = \mathrm{mgu}(t_1, t_1')$, if such mgu's exist.

We introduce below the symbolic evaluation relation $\downarrow_s$, which extends the evaluation relation to open terms. Intuitively, $\zeta \downarrow_\theta \eta$ means that ζ evaluates to η under any possible instance of θ. We require that $\zeta \downarrow_\theta \eta$ be *image-finite*, i.e., for each ζ, the set $\{(\theta, \eta) \mid \zeta \downarrow_\theta \eta\}$ is finite up to renaming of variables. The main advantage of the symbolic relation over the

[1] We assume the standard notion of composition of substitutions (cf. [25]): for $\theta_1 = [t_1/y_1, \ldots, t_k/y_k]$ and $\theta_2 = [t_1'/x_1, \ldots, t_n'/y_n]$, $\theta_1\theta_2 = [t_1\theta_2/y_1, \ldots, t_k\theta_2/y_k] \cup \{[t_i'/x_i] \in \theta_2 \mid x_i \notin \mathrm{dom}(\theta_1)\} \setminus \mathrm{Id}$, where Id_V is the identity relation on variables.

Table 4
Symbolic Evaluation Relation ($\downarrow_s$) for $\mathcal{F}^s_{pk}$

$$(\text{DEC}_\text{S})\quad \mathsf{dec}^{\mathsf{pk}}_{\zeta}(\eta) \overset{\theta}{\rightsquigarrow}_\text{S} x_1\theta \qquad \theta = \text{mgu}(\eta = \{\!|x_1|\!\}_{x_2^+}, \zeta = x_2^-)$$

$$(\text{PRJ}_\text{S})\quad \pi_i(\zeta) \overset{\theta}{\rightsquigarrow}_\text{S} x_i\theta \quad (i=1,2) \qquad \theta = \text{mgu}(\zeta, \langle x_1, x_2\rangle)$$

$$(\text{ENC}_\text{S})\quad \{\!|\zeta|\!\}_x \overset{\theta}{\rightsquigarrow}_\text{S} \{\!|\zeta\theta|\!\}_{x^+} \qquad \theta = [x^+/x]$$

$$(\text{CTX}_\text{S})\quad \frac{\zeta \overset{\theta}{\rightsquigarrow}_\text{S} \zeta'}{C[\zeta] \overset{\theta}{\rightsquigarrow}_\text{S} C\theta[\zeta']}$$

$$\text{SYMBOLIC EVALUATION}\quad \zeta \downarrow_\theta \eta \quad \text{iff} \quad \zeta \overset{\theta_1}{\rightsquigarrow}_\text{S} \cdots \overset{\theta_n}{\rightsquigarrow}_\text{S} \eta \ \text{ and } \ \theta = \theta_1 \cdots \theta_n$$

Variables x_1 and x_2 are fresh.

concrete one ($\downarrow$) is that infinitely many pairs (ζ, η) such that $\zeta \downarrow \eta$ can be represented by means of a single judgement $\zeta_0 \downarrow_\theta \eta_0$, for some ζ_0, θ, η_0.

Definition 5 (*Symbolic frame*). A *symbolic frame* is a pair $\mathcal{F}^s = (\mathcal{F}, \downarrow_s)$, where $\mathcal{F} = (\Sigma, \mathcal{M}, \downarrow)$ is a frame, and $\downarrow_s \subseteq \mathcal{E}_\Sigma \times \mathsf{Subst} \times \mathcal{E}_\Sigma$ is an image-finite *symbolic evaluation relation* (we write $\zeta \downarrow_\theta \eta$ for $(\zeta, \theta, \eta) \in \downarrow_s$) such that, for any expression ζ and ground substitution ρ with $\text{v}(\zeta) \subseteq \text{dom}(\rho)$, the following hold:

(a) If $\zeta\rho \downarrow \eta$, then there exist ξ, θ, ρ_0 such that $\zeta \downarrow_\theta \xi$, $\rho = (\theta\rho_0)_{|\text{dom}(\rho)}$ and $\eta = \xi\rho_0$. Furthermore, $\eta \in \mathcal{M}$ implies $\xi \in \mathcal{M}$.
(b) If $\zeta \downarrow_\theta \xi$ and $\rho = \theta\rho_0$, for some ρ_0, then $\zeta\rho \downarrow \xi\rho_0$.

Note that in the above definition, θ may in general both contain variables of ζ and introduce fresh variables.

Example 4 (*Public-Key Encryption—Cont.*). $\mathcal{F}^s_{pk}$ is defined as $(\mathcal{F}_{pk}, \downarrow_s)$, where $\downarrow_s$ is the reflexive and transitive closure of the relation $(\rightsquigarrow_\text{S})$, as given in Table 4.

Proposition 1. $\mathcal{F}^s_{pk}$ is a symbolic frame.

Proof. See Appendix C.1. □

We now come to symbolic counterparts of traces and configurations. Condition (b) in the definition below states that only the environment can introduce variables into symbolic traces.

Definition 6 (*Symbolic traces and configurations*). A *symbolic trace* is a string $s \in Act^*$ such that: (a) $\text{en}(s) = \emptyset$, and (b) for each s_1, s_2, α and x, if $s = s_1 \cdot \alpha \cdot s_2$ and

$x \in \mathrm{v}(\alpha) - \mathrm{v}(s_1)$ then α is an input action. Symbolic traces are ranged over by $\sigma, \sigma', \ldots$. A *symbolic configuration*, written $\langle \sigma,\ A \rangle_{\mathrm{S}}$, is a pair composed by a symbolic trace σ and an agent A, such that $\mathrm{en}(A) = \emptyset$ and $\mathrm{v}(A) \subseteq \mathrm{v}(\sigma)$.

Note that, due to Condition (b) in the Definition 6, for instance, $\overline{\mathsf{a}}\langle x^+ \rangle \cdot \mathsf{a}\langle \{\!|h|\!\}_{x^+} \rangle$ is not a symbolic trace, while $\mathsf{a}\langle \{\!|h|\!\}_{x^+} \rangle \cdot \overline{\mathsf{a}}\langle x^+ \rangle$ is.

Once a symbolic frame $\mathcal{F}^s$ is fixed, configurations can be equipped with a symbolic transition relation, $\longrightarrow_{\mathrm{S}}$, as defined by the rules in Table 5 (for the sake of readability we omit any explicit reference to $\mathcal{F}^s$). There, a function $\mathrm{new}(\cdot)$ is assumed such that, for any given $V \subseteq_{\mathrm{fin}} \mathcal{V}$, $\mathrm{new}(V)$ is a variable not in V. We also make use of the following notation: for $Y = \{y_1, \ldots, y_n\}$, by $Y = \mathrm{new}(V)$ we mean that $y_1 = \mathrm{new}(V)$, $y_2 = \mathrm{new}(V \cup \{y_1\}), \ldots, y_n = \mathrm{new}(V \cup \{y_1, \ldots, y_{n-1}\})$. Moreover, $\mathcal{C} \xrightarrow{\theta}_{\mathrm{S}} \mathcal{C}'$ stands for $\mathcal{C} \longrightarrow_{\mathrm{S}} \mathcal{C}'$, where θ is the substitution applied to $\mathcal{C}'$ in the reduction step, i.e., $\langle \sigma,\ A \rangle_{\mathrm{S}} \xrightarrow{\theta}_{\mathrm{S}} \langle \sigma',\ A' \rangle_{\mathrm{S}}$ means $\langle \sigma,\ A \rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma',\ A' \rangle_{\mathrm{S}}$ and $\sigma' = \sigma\theta \cdot \alpha$ or $\sigma' = \sigma\theta$, for some action α.

Note that, differently from the concrete semantics, input variables are *not* instantiated immediately (rule ($\textsc{Inp}_{\mathrm{S}}$). Rather, constraints on these variables are computed and propagated as soon as needed. This may occur due to rules ($\textsc{Out}_{\mathrm{S}}$), ($\textsc{Let}_{\mathrm{S}}$) and ($\textsc{Match}_{\mathrm{S}}$). In the following example, after the first step, variable x gets instantiated to name b by a ($\textsc{Match}_s$)-reduction:

$$\langle \varepsilon,\ \mathsf{a}(x).[x = b]P \rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \mathsf{a}\langle x \rangle,\ [x = b]P \rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \mathsf{a}\langle b \rangle,\ P[{}^{b}/_{x}] \rangle_{\mathrm{S}}.$$

Whenever $\langle \sigma,\ A \rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}}^{*} \langle \sigma',\ A' \rangle_{\mathrm{S}}$ for some A', we say that $\langle \sigma,\ A \rangle_{\mathrm{S}}$ *symbolically generates* σ', and write $\langle \sigma,\ A \rangle_{\mathrm{S}} \searrow_{\mathrm{S}} \sigma'$. Due to the image-finiteness of $\downarrow_\theta$, the relation $\longrightarrow_{\mathrm{S}}$ is finitely branching, hence each configuration generates a finite number of symbolic traces. For example, consider the process $P = \mathsf{a}(y).\,\mathsf{let}\, x{=}\mathsf{dec}_{k^-}^{\mathsf{pk}}(y)\mathsf{in}\,\overline{\mathsf{a}}\langle x \rangle.\,0$. By the rules in Table 4, the initial configuration $\langle \varepsilon,\ P \rangle_{\mathrm{S}}$ generates the following symbolic traces:

$$\varepsilon, \quad \mathsf{a}\langle y \rangle, \quad \mathsf{a}\langle \{\!|z|\!\}_{k^+} \rangle \text{ (for some fresh } z), \quad \mathsf{a}\langle \{\!|z|\!\}_{k^+} \rangle \cdot \overline{\mathsf{a}}\langle z \rangle.$$

It is important to stress that many symbolic traces are in fact 'inconsistent', that is, sequences of actions that cannot be instantiated to any concrete trace. For instance, the symbolic trace $\mathsf{a}\langle \{\!|z|\!\}_{k^+} \rangle \cdot \overline{\mathsf{a}}\langle z \rangle$ above is not relevant for the analysis, because the environment cannot generate the value k^+ in $\{\!|z|\!\}_{k^+}$ (i.e. $\varepsilon \not\vdash k^+$, hence $\varepsilon \not\vdash \{\!|z|\!\}_{k^+}$). The problem of detecting these inconsistent traces, that might give rise to 'false positives' when checking protocol properties, will be faced in the next section. The notion of consistency is formally defined below.

Definition 7. Given a symbolic trace σ and a ground substitution ρ, we say that ρ *satisfies* σ if $\sigma\rho$ is a trace. If it is the case, we also say that $\sigma\rho$ is a *solution* of σ. A symbolic trace σ is *consistent* if there exist solutions of σ.

The task of checking consistency of symbolic traces is a crucial point of the verification method presented in the next section. Theorem 1 below establishes a correspondence between the concrete and the symbolic transition relations.

Table 5
Rules for symbolic transition relation ($\longrightarrow_{\mathrm{S}}$)

$$(\textsc{Inp}_{\mathrm{S}}) \quad \langle \sigma,\ \mathsf{a}(x).A\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma\cdot\mathsf{a}\langle x\rangle,\ A\rangle_{\mathrm{S}}$$

$$(\textsc{Out}_{\mathrm{S}}) \quad \langle \sigma,\ \overline{\mathsf{a}}\langle\zeta\rangle.A\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma\theta\cdot\overline{\mathsf{a}}\langle M\rangle,\ A\theta\rangle_{\mathrm{S}} \quad \zeta\downarrow_{\theta} M$$

$$(\textsc{Let}_{\mathrm{S}}) \quad \langle \sigma,\ \mathsf{let}\ x{=}\zeta\ \mathsf{in}\ A\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma\theta,\ A\theta[\xi/x]\rangle_{\mathrm{S}} \quad \zeta\downarrow_{\theta}\xi$$

$$(\textsc{Match}_{\mathrm{S}}) \quad \langle \sigma,\ [\zeta=\eta]A\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma\theta,\ A\theta\rangle_{\mathrm{S}} \quad \zeta\downarrow_{\theta_1}\xi_1,\ \eta\,\theta_1\downarrow_{\theta_2}\xi_2,\ \theta_3=\mathrm{mgu}(\xi_1\theta_2,\xi_2),\ \theta=\theta_1\theta_2\theta_3$$

$$(\textsc{Par}_{\mathrm{S}}) \quad \frac{\langle \sigma,\ A\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma',\ A'\rangle_{\mathrm{S}}}{\langle \sigma,\ A\parallel B\rangle_{\mathrm{S}} \longrightarrow_{\mathrm{S}} \langle \sigma',\ A'\parallel B\rangle_{\mathrm{S}}}$$

plus symmetric version of ($\textsc{Par}_{\mathrm{S}}$). In the above rules it is assumed that, for V the set of free variables in the source configuration:
(i) $x = \mathrm{new}(V)$;
(ii) $\mathrm{v}(\theta)\setminus V = \mathrm{new}(V)$;
(iii) in rule ($\textsc{Par}_{\mathrm{S}}$), $B' = B\theta$ where $\langle \sigma,\ A\rangle_{\mathrm{S}} \xrightarrow{\theta}_{\mathrm{S}} \langle \sigma',\ A'\rangle_{\mathrm{S}}$;
(iv) $\mathrm{msg}(\sigma)\theta \subseteq \mathcal{M}$.

Theorem 1 (*Concrete vs. symbolic semantics*). *Let $\mathcal{F}^s$ be a symbolic frame, $\mathcal{C}$ be an initial configuration and s be a trace of $\mathcal{F}$. Then $\mathcal{C} \searrow s$ if and only if there exists σ such that $\mathcal{C} \searrow_{\mathrm{S}} \sigma$ and s is a solution of σ.*

Proof. See Appendix A. □

4. A verification method

We first define *regular frames*, i.e., frames for which it is possible to determine a finite *basis* function for the synthesis of messages. Then we introduce a *refinement* procedure that can be used to check consistency of symbolic traces. Finally, we present a verification method based on refinement that applies to regular frames.

4.1. Regular frames

It is convenient to extend the syntax of messages with a new class of variables. Informally, these variables will be used as place-holders for generic messages known to the environment. Formally, we consider a new set $\widehat{\mathcal{V}}$ of *marked* variables, in bijection with $\mathcal{V}$ via a mapping $\hat{\cdot}$; thus, variables $x, y, z, \ldots$ have marked counterparts $\hat{x}, \hat{y}, \hat{z}, \ldots$. Marked messages (resp., traces) are messages (resp., traces) that may also contain marked variables. Also, for

$S \subseteq \mathcal{M}$, the set $\mathcal{H}(S)$ in Definition 2 is extended to include marked variables, that is, we re-define $\mathcal{H}^0_{\mathcal{F}}(S)$ as follows:

$$\mathcal{H}^0_{\mathcal{F}}(S) \stackrel{\triangle}{=} S \cup \mathcal{EN} \cup \widehat{\mathcal{V}}.$$

The deduction relation ($S \vdash M$) remains formally unchanged. Note that in case S and M do not contain marked variables, the new definition coincides with Definition 2. Since marked variables are intended to carry messages known to the environment, the satisfaction relation is extended below to marked symbolic trace according to this intuition. For any $\hat{x}$ and any trace σ, we denote by $\sigma\backslash\hat{x}$ the longest prefix of σ not containing $\hat{x}$.

Definition 8 (*Consistency*). Let σ be a marked symbolic trace and ρ be a ground substitution. We say that ρ *satisfies* σ if $\sigma\rho$ is a trace and, for each $\hat{x} \in \mathrm{v}(\sigma)$, it holds that $(\sigma\backslash\hat{x})\rho \vdash \rho(\hat{x})$. In this case, we say that $\sigma\rho$ is a *solution* of σ, and that σ is consistent.

The terminology introduced above agrees with Definition 7 when σ does not contain marked variables. We give now the definition of solved form, that lifts the concept of trace to the non-ground case (note that this definition is formally the same as the definition of trace, Definition 3).

Definition 9 (*Solved forms*). Let σ be a marked symbolic trace. We say σ is in *solved form (sf)* if for every σ_1, $\mathsf{a}\langle M\rangle$ and σ_2 s.t. $\sigma = \sigma_1 \cdot \mathsf{a}\langle M\rangle \cdot \sigma_2$ it holds that $\sigma_1 \vdash M$.

Next, we define regular frames, which enjoy a 'finite-basis' property. Basically, this property states the existence, for any σ in solved form, of a finite set of 'building blocks', out of which all messages deducible from σ can be syntactically built: this requirement is stated by Condition 1, below. Condition 2 requires that basis functions and substitutions commute with each other. In fact, this would be the exact meaning of $\mathbf{b}(\sigma\rho) = \mathbf{b}(\sigma)\rho$, but for our purposes inclusion $\subseteq$ suffices. Define the set of messages deducible from σ as $\mathcal{D}(\sigma) \stackrel{\triangle}{=} \{M \mid \sigma \vdash M\}$. The additional 'sanity' conditions $\mathrm{v}(\mathbf{b}(\sigma)) \subseteq \mathrm{v}(\sigma)$ and $\mathbf{b}(\sigma) \subseteq \mathcal{D}(\sigma) \setminus (\mathcal{EN} \cup \widehat{\mathcal{V}})$ are also desirable to rule out weird or redundant bases (in fact, they are sufficient to guarantee correctness of our method, as we shall see in Section 4).

Definition 10 (*Regular frames*). A symbolic frame $\mathcal{F}^s$ is *regular* if there exists a *basis function* $\mathbf{b} : Act^* \longrightarrow \mathcal{P}_f(\mathcal{M})$ such that for each solved form σ of $\mathcal{F}^s$, $\mathrm{v}(\mathbf{b}(\sigma)) \subseteq \mathrm{v}(\sigma)$ and $\mathbf{b}(\sigma) \subseteq \mathcal{D}(\sigma) \setminus (\mathcal{EN} \cup \widehat{\mathcal{V}})$ and for all ρ satisfying σ:
(1) $\sigma\rho \vdash M$ if and only if $M \in \mathcal{H}(\mathbf{b}(\sigma\rho))$;
(2) $\mathbf{b}(\sigma\rho) \subseteq \mathbf{b}(\sigma)\rho$.
For each σ, $\mathbf{b}(\sigma)$ is said a *basis* of σ.

Example 5 (*Public-key encryption—Cont.*). Let us consider the frame for public key encryption introduced in Example 1. A basis function for $\mathcal{F}^s_{pk}$ selects, for a given σ in sf, a set consisting of plain variables, local names, keys, and encrypted messages that cannot decomposed out of smaller messages deducible from σ. In the following, by $M = (u)^{\pm}$ we mean $M = u$ or $M = u^+$ or $M = u^-$.

Definition 11 (*Function* $\mathbf{b}_{pk}$).

$$\mathbf{b}_{pk}(\sigma) \stackrel{\triangle}{=} \{\, M \mid (\sigma \vdash M) \text{ and } (M = (u)^{\pm}, \text{ for some } u \in \mathcal{LN} \cup \mathcal{V}, \text{ or } \exists\, N, u : M = \{\!|N|\!\}_{u^+} \text{ and } \sigma \not\vdash \langle N, u^+ \rangle\ )\ \}.$$

Note that $\mathbf{b}_{pk}(\sigma)$ may in general contain encrypted open terms, e.g. for $\sigma = \mathsf{c}\langle \hat{x} \rangle \cdot \overline{\mathsf{c}}\langle \{\!|\hat{x}|\!\}_{k^+} \rangle$, $\mathbf{b}_{pk}(\sigma) = \{\{\!|\hat{x}|\!\}_{k^+}\}$. In practice, for a given σ in sf, the set $\mathbf{b}_{pk}(\sigma)$ can be effectively computed by an iterative procedure, which repeatedly applies destructors ($\mathsf{dec}^{\mathsf{pk}}_{(\cdot)}(\cdot)$ and $\pi_i(\cdot)$) to messages in σ, until some fixed point is reached. This procedure always terminates.

The frame $\mathcal{F}^s_{pk}$ defined in the above example is regular, as stated by the following theorem, whose proof is reported in Appendix C.2.

Theorem 2. *$\mathcal{F}^s_{pk}$ is a regular frame with basis function $\mathbf{b}_{pk}$.*

We shall exhibit an example of symbolic but non-regular frame in Section 5.

4.2. Refinement

In the refinement procedure, each input message in a symbolic trace is tentatively unified to some message that can be synthesized from a basis for past messages. By iterating this step, one can check whether a given symbolic trace can eventually be instantiated to a trace in the concrete model. In particular, given any symbolic trace σ we can compute the set of the 'most general instances' of σ satisfying the solved form property, denoted by $\mathbf{SF}(\sigma)$.

Definition 12 (*Refinement and* $\mathbf{SF}(\sigma)$). We let *refinement*, written $\succ$, be the least binary relation over marked symbolic traces of a regular frame given by the two rules below. In (REF$_1$), σ' is the longest prefix of σ that is in solved form and $\sigma = \sigma' \cdot \mathsf{a}\langle M \rangle \cdot \sigma''$, for some σ''. Assume $N, N' \notin \mathcal{V} \cup \widehat{\mathcal{V}}$.

$$(\text{REF}_1)\ \frac{M = C[N] \quad N' \in \mathbf{b}(\sigma') \quad \theta = \mathrm{mgu}(N, N')}{\sigma \ \succ\ \sigma\theta\theta_0}\ ,$$

$$(\text{REF}_2)\ \frac{x \in \mathrm{v}(M)}{\sigma \ \succ\ \sigma[\hat{x}/x]}\ ,$$

where $\theta \neq \varepsilon$, $\theta_0 \stackrel{\triangle}{=} \{\, {}^{x}/_{\hat{x}} \mid \hat{x} \in \mathrm{v}(\sigma) \text{ and } |(\sigma\theta)\backslash\hat{x}| < |\sigma\backslash\hat{x}| \,\}$.

For any symbolic trace σ, we let $\mathbf{SF}(\sigma) \stackrel{\triangle}{=} \{\, \sigma' \mid \sigma \ \succ^*\ \sigma' \text{ and } \sigma' \text{ is in } sf \,\}$.

Rule (REF$_1$) implements the basic step of refinement: a subterm N of M gets unified, via θ, to an element of $\mathbf{b}(\sigma')$. By rule (REF$_2$) a variable can get marked: it will be treated as a known constant in subsequent steps of refinement. Note that in a (REF$_1$)-step a marked variable $\hat{x}$ may possibly be'unmarked' back to the plain variable x. This is achieved via the renaming θ_0, and happens precisely when application of θ causes the first occurrence of $\hat{x}$ to move backward in the trace.

Example 6. Consider $\sigma = \overline{\mathsf{c}}\langle\{\!|a|\!\}_{k^+}, d\rangle \cdot \overline{\mathsf{c}}\langle\{\!|b|\!\}_{k^+}, e\rangle \cdot \mathsf{c}\langle\{\!|x|\!\}_{k^+}, d\rangle$, and let $\sigma' = \overline{\mathsf{c}}\langle\{\!|a|\!\}_{k^+}, d\rangle \cdot \overline{\mathsf{c}}\langle\{\!|b|\!\}_{k^+}, e\rangle$. It holds that

$$\mathbf{b}_{pk}(\sigma') = \{\{\!|a|\!\}_{k^+},\ d,\ \{\!|b|\!\}_{k^+},\ e\}.$$

Two possible refinements of σ, via the first rule, are $\sigma \succ \sigma[{}^{a}/_{x}]$ and $\sigma \succ \sigma[{}^{b}/_{x}]$; the refined traces are in sf. The remaining refinement of σ is $\sigma \succ \sigma[{}^{\hat{x}}/_{x}]$. Note that $\sigma[{}^{\hat{x}}/_{x}]$ is not in sf, since $\sigma' \not\vdash \langle\{\!|\hat{x}|\!\}_{k^+}, d\rangle$ (in particular, $\sigma' \not\vdash \{\!|\hat{x}|\!\}_{k^+}$). Hence $\mathbf{SF}(\sigma) = \{\sigma[{}^{a}/_{x}],\ \sigma[{}^{b}/_{x}]\}$.

Proposition 2. *Let σ be a symbolic trace. Then $\mathbf{SF}(\sigma)$ is finite.*

Proof. The thesis follows from two facts: (a) $\succ$ is a finitely branching relation, and (b) infinite sequences of refinement steps cannot arise. As to the latter point, first note that each (REF$_1$)-step eliminates at least one variable: this stems from the definition of mgu, and from the fact that $\mathrm{v}(N, N') \subseteq \mathrm{v}(\sigma)$. Hence any sequence of refinement steps can contain only finitely many (REF$_1$)-steps and, after the last of them, rule (REF$_2$) can only be applied a finite number of times. □

We now prove that the solutions of a symbolic trace σ can be completely characterised in terms of the solutions of the symbolic traces in $\mathbf{SF}(\sigma)$. The proof of this fact is based on Lemma 1 below, which basically states that any consistent symbolic trace that is not in solved form can be further refined.

Lemma 1 (*Progression lemma*). *Let $\mathcal{F}^s$ be a regular frame and σ be a marked symbolic trace. Suppose that there exists some ρ which satisfies σ. Then either σ is in sf, or there are σ' and ρ' such that $\sigma \succ \sigma'$, $\sigma\rho = \sigma'\rho'$, and ρ' satisfies σ'.*

Proof. See Appendix B. □

Theorem 3. *Let $\mathcal{F}^s$ be a regular frame, σ be a symbolic trace, and s be a trace in $\mathcal{F}^s$. Then s is a solution of σ if and only if s is a solution of some $\sigma' \in \mathbf{SF}(\sigma)$.*

Proof. Suppose $s = \sigma'\rho$ is a solution of σ'. Then, obviously s is a solution of σ, as $\sigma' = \sigma\theta$, for some θ (note that by definition σ does not contain marked variables). On the converse, suppose $s = \sigma\rho$ is a solution of σ. By repeated application of the previous lemma, we find that there is σ' in solved form and ρ' s.t. that $\sigma \succ^* \sigma'$, $s = \sigma\rho = \sigma'\rho'$ and ρ' satisfies σ'. Hence, by definition, s is a solution of σ'. □

Note that each solved form σ has a non-empty set of solutions: a trivial solution is obtained by mapping each variable of σ to any name in $\mathcal{EN}$. This fact and the above theorem imply that a symbolic trace σ is consistent if and only if $\mathbf{SF}(\sigma) \neq \emptyset$. It follows that computing $\mathbf{SF}(\sigma)$ gives an effective method to decide consistency of σ.

Table 6
The verification method

M$(\mathcal{C}, \alpha \hookleftarrow \beta)$
1. compute $\mathbf{Mod}_{\mathcal{C}} = \{\sigma \mid \mathcal{C} \searrow_{\mathrm{S}} \sigma\}$;
2. **foreach** $\sigma \in \mathbf{Mod}_{\mathcal{C}}$ **do**
3. **foreach** action γ in σ **do**
4. **if** $\exists\, \theta = \mathrm{mgu}(\beta, \gamma)$ **and**
5. $\exists\, \sigma' \in \mathbf{SF}(\sigma\theta)$ **where** $\sigma' = \sigma\theta\theta'$ **and**
6. $\alpha\theta\theta'$ does not occur prior to $\beta\theta\theta'$ in σ'
7. **then return**(No, σ');
8. **return**(Yes);

4.3. The verification method

The method $\mathbf{M}(\mathcal{C}, \alpha \hookleftarrow \beta)$ described in Table 6 can be used to verify if $\mathcal{C} \vDash \alpha \hookleftarrow \beta$ or not. If the property is not satisfied, the method computes a trace violating the property, that is, an attack on $\mathcal{C}$. The method always terminates, because the symbolic transition relation $\longrightarrow_{\mathrm{S}}$ is finitely branching, hence the set $\mathbf{Mod}_{\mathcal{C}}$ of symbolic traces generated by $\mathcal{C}$ is finite.

The functioning of the method is best explained by considering the specific case $\alpha = \bot$, i.e. verification of $\mathcal{C} \vDash \bot \hookleftarrow \beta$. This means verifying that in the *concrete* semantics, no instance of action β is ever executed starting from $\mathcal{C}$. By the correspondence between symbolic and concrete semantics (Theorem 1), this amounts to checking that for each σ symbolically generated by $\mathcal{C}$, no solution of σ contains an instance of β. The method proceeds as follows. First, it checks whether there is a mgu θ of γ and β, for every action γ of $\sigma \in \mathbf{Mod}_{\mathcal{C}}$. If, for every σ, such a θ does not exist, or it exists but $\sigma\theta$ is not consistent (this means that the check $\exists\, \sigma' \in \mathbf{SF}(\sigma\theta)$ at step 5 fails), then the property holds true, otherwise it does not, and the trace σ' violating the property is reported.

We shall see a step-by-step illustration of our method at work on a specific example in Section 6.

Remark 1. In practice, rather than generating the whole set of symbolic traces at once (step 1) and then checking the property, it is convenient to work 'on-the-fly' and comparing every last symbolic action γ taken by the configuration against action β of the property $\alpha \hookleftarrow \beta$; the refinement procedure $\mathbf{SF}(\cdot)$ is invoked only when β and γ are unifiable. This is, in fact, the way our symbolic trace analyser STA works. The complexity of the method in the worst case is expected to be exponential, since the analysis problem is easily seen to be NP-hard (see e.g. [38]).

The correctness and completeness of the method in the general case is stated by Theorem 4 below.

Theorem 4 (*Correctness and completeness*). *Let $\mathcal{F}^s$ be a regular frame, $\mathcal{C}$ be an initial configuration of $\mathcal{F}^s$ and α and β be actions of $\mathcal{F}^s$ such that* $\mathrm{v}(\alpha) \subseteq \mathrm{v}(\beta)$.

(1) *If* $\mathbf{M}(\mathcal{C}, \alpha \leftrightarrow \beta)$ *returns* (No, σ') *then* $\mathcal{C} \not\vdash \alpha \leftrightarrow \beta$. *In particular, for any injective ground substitution* $\rho : \mathrm{v}(\sigma') \to \mathcal{EN}$, *it holds that* $\mathcal{C} \searrow (\sigma'\rho)$ *and* $(\sigma'\rho) \not\vdash \alpha \leftrightarrow \beta$.
(2) *If* $\mathcal{C} \not\vdash \alpha \leftrightarrow \beta$ *then* $\mathbf{M}(\mathcal{C}, \alpha \leftrightarrow \beta)$ *returns* (No, σ') *and for any injective ground substitution* $\rho : \mathrm{v}(\sigma') \to \mathcal{EN}$, $\mathcal{C} \searrow (\sigma'\rho)$ *and* $(\sigma'\rho) \not\vdash \alpha \leftrightarrow \beta$.

Proof. (1) According to the method, there exist a trace σ, an action $\gamma \in \sigma$, a unifier $\theta = \mathrm{mgu}(\gamma, \beta)$, and θ' such that: $\sigma' = \sigma\theta\theta' \in \mathbf{SF}(\sigma\theta)$, $\mathcal{C} \searrow_{\mathrm{S}} \sigma$, and

$$\beta\theta\theta' \text{ occurs in } \sigma' \text{ but } \alpha\theta\theta' \text{ does not occur prior to } \beta\theta\theta' \text{ in } \sigma'. \tag{1}$$

Given ρ as in the hypotheses, $s \stackrel{\triangle}{=} \sigma'\rho$ is a solution of σ and, by applying Corollary 1, it follows that $\mathcal{C} \searrow s$. Also, by (1) and by the injectivity of ρ on $\mathrm{v}(\sigma')$, $\alpha\theta\theta'\rho$ may not occur prior to $\beta\theta\theta'\rho$ in s, otherwise $\alpha\theta\theta'\rho\rho^{-1} = \alpha\theta\theta'$ would occur prior to $\beta\theta\theta'\rho\rho^{-1} = \beta\theta\theta'$ in σ', contrary to (1). Hence $s \not\vdash \alpha \leftrightarrow \beta$.

(2) Suppose $\mathcal{C} \searrow s$ and $s \not\vdash \alpha \leftrightarrow \beta$. By definition, there exists ρ_1 such that $\alpha\rho_1$ does not occur prior to $\beta\rho_1$ in s. Let i be the position of the leftmost occurrence of $\beta\rho_1$ in s. By $\mathcal{C} \searrow s$ and by Corollary 1, it follows that there exists σ such that $\mathcal{C} \searrow_{\mathrm{S}} \sigma$ and s is a solution of σ, i.e. $s = \sigma\rho_2$, for some ρ_2. Since $\mathrm{v}(\alpha)$ and $\mathrm{v}(\beta)$ are universally quantified, we can assume $\mathrm{v}(\alpha, \beta) \cap \mathrm{v}(\sigma) = \emptyset$ and so $\mathrm{dom}(\rho_1) \cap \mathrm{dom}(\rho_2) = \emptyset$. Let $\rho_0 \stackrel{\triangle}{=} \rho_1 \cup \rho_2$ and γ be the ith element of σ. It follows that $\gamma\rho_0 = \beta\rho_0$. Thus, we can consider $\theta = \mathrm{mgu}(\beta, \gamma)$. By definition, ρ_0 is an instance of θ, i.e. $\rho_0 = \theta\rho_0'$, for some ρ_0'. Hence, $s = \sigma\theta\rho_0'$ is a solution of $\sigma\theta$. By Theorem 3, there exists a sf $\sigma' = \sigma\theta\theta' \in \mathbf{SF}(\sigma\theta)$ such that s is a solution of σ', i.e. $s = \sigma'\rho_0''$, for some ρ_0''. Necessarily

$$\alpha\theta\theta' \text{ does not occur prior to } \gamma\theta\theta' \text{ in } \sigma', \tag{2}$$

because otherwise $\alpha\theta\theta'\rho_0'' = \alpha\rho_1$ would occur prior to $\gamma\theta\theta'\rho_0'' = \beta\rho_1$ in s, contradicting the hypotheses. Therefore, we have found σ, γ, θ, and σ' such that $\mathbf{M}(\mathcal{C}, \alpha \leftrightarrow \beta)$ returns (No, σ'). Finally, given ρ as in the hypotheses, $\sigma'\rho$ is a solution of σ. Thus, $\mathcal{C} \searrow (\sigma'\rho)$ by the fact that $\mathcal{C} \searrow_{\mathrm{S}} \sigma$ and by Corollary 1. Also, $(\sigma'\rho) \not\vdash \alpha \leftrightarrow \beta$ as $\beta\theta\theta'\rho$ $(= \gamma\theta\theta'\rho)$ occurs in $\sigma'\rho$ at position i, but $\alpha\theta\theta'\rho$ does not occur in $\sigma'\rho$ prior to position i, by (2) and by the injectivity of ρ. □

Note that assertion (1) (correctness) of the above theorem only depends on the properties of the set $\mathbf{SF}(\sigma)$ of (a) being finite, and (b) containing only instances of σ. These two properties depends entirely on the sanity conditions of the definition of basis function. Thus, it makes sense to weaken Definition 10 as follows:

Definition 13. A symbolic frame is *weakly regular* if for each solved form σ it holds that $\mathrm{v}(\mathbf{b}(\sigma)) \subseteq \mathrm{v}(\sigma)$ and $\mathbf{b}(\sigma) \subseteq \mathcal{D}(\sigma) \setminus (\mathcal{EN} \cup \widehat{\mathcal{V}})$.

This allows us to state a useful and more general form of correctness:

Theorem 5. *Let* $\mathcal{F}^s$ *be a weakly regular frame,* $\mathcal{C}$ *be an initial configuration of* $\mathcal{F}^s$ *and* α *and* β *be actions of* $\mathcal{F}^s$ *such that* $\mathrm{v}(\alpha) \subseteq \mathrm{v}(\beta)$. *Then assertion* (1) *stated in Theorem* 4 *holds true.*

5. 'Black-box' cryptographic primitives

We consider extending the symbolic frame for public key cryptography $\mathcal{F}^s_{pk}$ to deal with some of the most common cryptographic operations.

The set Σ is enriched by means of appropriate operators for shared-key encryption $\{\cdot\}_{(\cdot)}$ and decryption $\mathsf{dec}^{\mathsf{sk}}_{(\cdot)}(\cdot)$, digital signing $[\![\cdot]\!]_{(\cdot)}$ and verifying $\mathsf{dec}^{\mathsf{ds}}_{(\cdot)}(\cdot)$ and hashing $H(\cdot)$. The syntax of messages is extended via the following additional clauses:

$$M, N ::= \quad ... \quad \text{as in Table 1}$$
$$\mid \{M\}_u \mid [\![M]\!]_{u^-} \mid H(M)\,.$$

The symbolic and concrete evaluations are given in terms of an auxiliary relation $\rightsquigarrow$, defined as expected. In particular, hashing has no rules, digital signature rules are just the same as for public key, but the roles of u^+ and u^- are swapped. For shared key, the concrete and symbolic rules are as follows:

$$\mathsf{dec}^{\mathsf{sk}}_\eta(\{\zeta\}_\eta) \rightsquigarrow \zeta \qquad \mathsf{dec}^{\mathsf{sk}}_\eta(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} x\theta \quad \text{where } \theta = \mathrm{mgu}(\zeta = \{x\}_\eta).$$

Pursuing the idea of selecting 'building blocks' out of deducible messages, a basis function for this frame can be defined by extending the basis function of the public key frame with all messages of the form $\{M\}_u$ (resp. $[\![M]\!]_{u^-}, H(M)$) such that $\sigma \not\vdash u$ (resp. $\sigma \not\vdash \langle M, u^-\rangle$, $\sigma \not\vdash M$).

The example below shows that the restriction to atomic keys is crucial to ensure the regularity condition.

Example 7 (*A non-regular frame*). Consider the frame defined above, but modified so as to allow messages with non-atomic keys in shared-key encryption, thus:

$$M, N ::= \cdots \mid \{M\}_N.$$

This frame is symbolic, but not regular. To see the latter, assume by contradiction there is a basis function $\mathbf{b}(\cdot)$ for this frame and consider the symbolic trace

$$\sigma = \overline{\mathsf{a}}\langle b\rangle \cdot \mathsf{a}\langle \hat{x}\rangle \cdot \overline{\mathsf{a}}\langle \{b\}_k\rangle \cdot \overline{\mathsf{a}}\langle \{c\}_{\{\hat{x}\}_k}\rangle$$

which is in solved form. Take $\rho = [^b/_{\hat{x}}]$. Clearly ρ satisfies σ, and $\sigma\rho \vdash c$. However, $c \notin \mathcal{H}(\mathbf{b}(\sigma\rho))$, which violates Condition 1 in the definition of basis function. To see this, note that by definition $\hat{x}, c \notin \mathbf{b}(\sigma)$, hence $c \notin \mathbf{b}(\sigma)\rho$, hence, by Condition 2, $c \notin \mathbf{b}(\sigma\rho)$. From this it easily follows by induction that $c \notin \mathcal{H}(\mathbf{b}(\sigma\rho))$.

6. Diffie–Hellman key exchange

In this section we instantiate our framework to the analysis of protocols based on a 'low-level' operation, modular exponentiation. First, we briefly recall one such protocol, the Diffie–Hellman key exchange, then we introduce a frame for shared-key and modular exponentiation, within which this protocol can be analysed.

The Diffie–Hellman protocol is intended for exchange of a secret key over an insecure medium, without prior sharing of any secret. The protocol has two public parameters: a large prime p and a generator α for the multiplicative group $\mathcal{Z}_p^* = \{1, \ldots, p-1\}$. Assuming A and B want to establish a shared secret key, they proceed as follows. First, A generates a random private value $n_A \in \mathcal{Z}_p^*$ and B generates a random private value $n_B \in \mathcal{Z}_p^*$. Next, A and B exchange their public values ($\mathsf{exp}(x, y)$ denotes $x^y \bmod p$):

1. $A \longrightarrow B : \mathsf{exp}(\alpha, n_A)$,
2. $B \longrightarrow A : \mathsf{exp}(\alpha, n_B)$.

Finally, A computes the key as $K = \mathsf{exp}(\mathsf{exp}(\alpha, n_B), n_A) = \mathsf{exp}(\alpha, n_A \times n_B)$, and B computes the key as $K = \mathsf{exp}(\mathsf{exp}(\alpha, n_A), n_B) = \mathsf{exp}(\alpha, n_A \times n_B)$. Now A and B share the value K, and A can use it to, say, encrypt a secret datum d and send it to B:

3. $A \longrightarrow B : \{d\}_K$.

The protocol's security depends on the difficulty of the discrete logarithm problem: it is computationally unfeasible to compute y if only x and $\mathsf{exp}(x, y)$ are known. The protocol is believed to be secure in the absence of 'active' attackers, but a well-known attack exists in the presence of active intruders.

Definition 14 (*Frame* $\mathcal{F}_{\mathrm{DH}}$). A frame for exponentiation and shared-key cryptography $\mathcal{F}_{\mathrm{DH}} = (\Sigma, \mathcal{M}, \downarrow)$ is defined in Table 7.

Besides shared-key encryption $\{\zeta\}_\eta$ and decryption $\mathsf{dec}_\eta^{\mathsf{sk}}(\zeta)$ (with η used as a key), the other symbols of Σ represent arithmetic operations modulo a fixed and public prime number, which is kept implicit. In particular, we have exponentiation $\mathsf{exp}(\zeta, \eta)$, root extraction $\mathsf{root}(\zeta, \eta)$, a constant α that represents a public generator, two symbols for multiplicative unit (unit, 1), two symbols for product $\mathsf{mult}(\zeta, \eta)$ and its result $\zeta \times \eta$, three symbols, $\mathsf{inv}(\zeta)$, $\mathsf{inv}'(\zeta)$ and ζ^{-1}, for the multiplicative inverse operation. The aim of using multiple symbols for each of the above operations is to ensure termination of the symbolic relation, as explained later on. All the underlying operations are computationally feasible. [2] A message is either a product of up to l values, for a fixed constant l, or a key or a message encrypted under a key. A key can be either an atomic object, or an exponential with base α and a product exponent ($\mathsf{exp}(\alpha, F)$).

Evaluation ($\downarrow$) is the reflexive and transitive closure of an auxiliary relation $\rightsquigarrow$. There, we use $\zeta_1 \times \zeta_2 \times \cdots \times \zeta_n$ as a shorthand for $\zeta_1 \times (\zeta_2 \times \cdots \times \zeta_n)$, while $(i_1, \ldots, i_n)$ is any permutation of $(1, \ldots, n)$. The relation $\rightsquigarrow$ is terminating, but not confluent. In fact, the non-determinism of $\rightsquigarrow$ is intended to model the commutativity and the associativity of the product operation, as reflected in the rule (MULT). Also, note rule (ROOT): in modular arithmetic, taking the kth root amounts to raising to $k^{-1} \bmod p-1$.

The choice of the above message formats and rules corresponds to imposing the following restrictions on the attacker and on the honest participants:

(1) there is a fixed upper bound (l) on the number of factors;

[2] An abstraction we make is that a unique operation is used to model both inverse mod p and inverse mod $p-1$ (the latter operation arises only inside exponents). Also, we are ignoring that, in modular arithmetic mod n, the inverse of k modulo n is defined only if $\gcd(k, n) = 1$ (see e.g. [29]).

Table 7
$\mathcal{F}_{\mathrm{DH}}$, a frame for modular exponentiation

$$\text{SIGNATURE} \quad \Sigma = \{\, \alpha,\ \mathsf{unit},\ 1,\ \{\cdot\}_{(\cdot)},\ \mathsf{dec}^{\mathsf{sk}}_{(\cdot)}(\cdot),\ \mathsf{exp}(\cdot,\cdot),\ \mathsf{root}(\cdot,\cdot),\ \cdot\times\cdot,\ \mathsf{mult}(\cdot,\cdot),\ \mathsf{inv}(\cdot),\ \mathsf{inv}'(\cdot),\ (\cdot)^{-1}\,\}$$

$$\text{FACTORS} \quad f ::= u \mid u^{-1}$$

$$\text{PRODUCTS} \quad F ::= 1 \mid f_1 \times \cdots \times f_k$$

$$\text{KEYS} \quad K, H ::= f \mid \mathsf{exp}(\alpha, F)$$

$$\text{MESSAGES} \quad M, N ::= F \mid K \mid \{M\}_K$$

$$\text{(DEC)}\ \mathsf{dec}^{\mathsf{sk}}_{\eta}(\{\zeta\}_{\eta}) \rightsquigarrow \zeta$$

$$\text{(MULT)}\ \mathsf{mult}(\zeta_1 \times \cdots \times \zeta_k, \zeta_{k+1} \times \cdots \times \zeta_n) \rightsquigarrow \zeta_{i_1} \times \cdots \times \zeta_{i_n} \qquad 1 \le k < n \le l$$

$$\text{(INV}_1\text{)}\ \mathsf{inv}(\zeta_1 \times \cdots \times \zeta_n) \rightsquigarrow \mathsf{inv}'(\zeta_1) \times \cdots \times \mathsf{inv}'(\zeta_n) \qquad n \le l$$

$$\text{(INV}_2\text{)}\ \mathsf{inv}'(\zeta^{-1}) \rightsquigarrow \zeta \qquad \text{(INV}_3\text{)}\ \mathsf{inv}'(\zeta) \rightsquigarrow \zeta^{-1} \qquad \text{(INV}_4\text{)}\ \mathsf{inv}'(\zeta) \times \zeta \rightsquigarrow \mathsf{unit}$$

$$\text{(UNIT}_1\text{)}\ \mathsf{unit} \times \zeta \rightsquigarrow \zeta \qquad \text{(UNIT}_2\text{)}\ \mathsf{unit} \rightsquigarrow 1$$

$$\text{(EXP)}\ \mathsf{exp}(\mathsf{exp}(\xi, \eta), \zeta) \rightsquigarrow \mathsf{exp}(\xi, \mathsf{mult}(\eta, \zeta))$$

$$\text{(ROOT)}\ \mathsf{root}(\mathsf{exp}(\xi, \eta), \zeta) \rightsquigarrow \mathsf{exp}(\xi, \mathsf{mult}(\eta, \mathsf{inv}(\zeta)))$$

$$\text{(CTX)}\ \frac{\zeta \rightsquigarrow \zeta'}{C[\zeta] \rightsquigarrow C[\zeta']} \qquad \text{EVALUATION} \quad \zeta \downarrow \eta \quad \text{iff} \quad \zeta \rightsquigarrow^{*} \eta$$

(2) product and inverse operations cannot be applied to exponentials and to encrypted terms;
(3) exponentiation starts from the base α, and exponents can only be product terms.

More accurately, starting from a term obeying the above conditions, an attacker is capable of 'deducing' all—though *not necessarily only*—AC variants of the message represented by the term. Thus, if one such variant, in a computation, leads to an attack, it will be considered by the operational model. Restriction (1) might be relaxed at the cost of introducing a set of mult_l operations, one for each $l \geqslant 0$, but for simplicity here we stick to the above model.

The example below illustrates typical usage of the evaluation and of the deduction relations.

Example 8. Consider a set $S = \{\, n_A, \mathsf{exp}(\alpha, n_B) \,\}$. Then, $S \vdash \mathsf{exp}(\alpha, n_A \times n_B)$ and $S \vdash \mathsf{exp}(\alpha, n_B \times n_A)$. A further example involves root extraction. Consider a set

$S = \{\, \{m\}_{\mathsf{exp}(\alpha,k\times l)},\ \mathsf{exp}(\alpha, k\times h),\ h,\ l\,\}$. Then, $S \vdash m$ since there exists $\zeta \in \mathcal{H}(S)$, $\zeta = \mathsf{dec}^{\mathsf{sk}}_{\eta}(\{m\}_{\mathsf{exp}(\alpha,k\times l)})$, with $\eta = \mathsf{exp}(\mathsf{root}(\mathsf{exp}(\alpha, k\times h), h), l)$, s.t. $\zeta \downarrow m$.

The symbolic evaluation relation $\downarrow_s$ of $\mathcal{F}_{\mathrm{DH}}$ is presented in Table 8: it is defined as the reflexive and transitive closure of the relation $\overset{\theta}{\rightsquigarrow}_{\mathrm{S}}$. We can now explain the adoption of multiple symbols in the case of product (mult and $\times$), inverse (inv, inv′ and $()^{-1}$), and unit (unit and 1). If we used just one symbol for, say, product, the rewrite rule ($\textsc{Mult}_S$) in Table 8 would be non-terminating, e.g.:

$$x \times y \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} (x_{i_1} \times \cdots \times x_{i_n}) \overset{\theta'}{\rightsquigarrow}_{\mathrm{S}} ((y_{j_1} \times \cdots \times y_{j_m}) \times \cdots \times x_{i_n}) \overset{\theta''}{\rightsquigarrow}_{\mathrm{S}} \cdots;$$

similarly, for inverse and unit operations. On the contrary, the use of multiple symbols and the form of the rules ensure termination of the evaluation relation.

Example 9. Consider $P = \overline{\mathsf{a}}\langle k\rangle.\, \mathsf{a}(x).\, \mathsf{let}\, z{=}\mathsf{root}(x, k)\mathsf{in}\, P'$. After an output action and an input action, the symbolic evaluation of $\mathsf{root}(x, k)$ produces a global substitution $\theta = [^{\mathsf{exp}(\alpha,\, x_1)}/_x]$ (x_1 fresh), to be applied to the whole configuration, and a local substitution $\theta' = [^{\mathsf{exp}(\alpha,\, x_1 \times k^{-1})}/_z]$, to be applied to $P'\theta$. I.e.

$$\langle \varepsilon,\ P\rangle_{\mathrm{S}} \longrightarrow^*_{\mathrm{S}} \langle \sigma\theta,\ P'\theta\theta'\rangle_{\mathrm{S}} \quad \text{with } \sigma = \overline{\mathsf{a}}\langle k\rangle \cdot \mathsf{a}\langle x\rangle.$$

In analogy to the public- and shared-key cases, we can define a basis function for $\mathcal{F}_{\mathrm{DH}}$ by considering all 'non-decomposable' messages deducible from a given σ: we have two more cases to consider here, non-decomposable products (i.e., products with no deducible sub-products), and exponentials with non-decomposable exponent. Let us write $G \subset F$ if G and F are products, and the set of factors of G is strictly included in F's.

Definition 15 (*A basis function for* $\mathcal{F}_{\mathrm{DH}}$). For each symbolic trace σ:

$$\begin{aligned}\mathbf{b}_{\mathrm{DH}}(\sigma) = \{M \mid (\sigma \vdash M) \text{ and } \big(\ & M \in \mathcal{LN} \cup \{\alpha, 1\} \cup \mathcal{V}\\ & \text{or } (M = F \text{ and } \sigma \nvdash G,\ \forall\, G \subset F)\\ & \text{or } (M = \mathsf{exp}(\alpha, F) \text{ and } \sigma \nvdash G,\ \forall\, G \subset F))\\ & \text{or } (M = \{M\}_K \text{ and } \sigma \nvdash K)\ \big)\ \}.\end{aligned}$$

We strongly conjecture that $\mathcal{F}_{\mathrm{DH}}$ equipped with the above basis function is a regular frame, but the details remain to be worked out. On the other hand, it is easy to check that this basis function turns $\mathcal{F}_{\mathrm{DH}}$ into a weakly regular frame (Definition 13). Thus we can appeal to Theorem 5 to make attacks found with the symbolic method correspond to attacks on the concrete model.

We now analyse the Diffie–Hellman Protocol. The process P defined below is a description of the Diffie–Hellman protocol presented in the introduction. For simplicity, we just describe a one-session version of the protocol, using again a few obvious

Table 8
Symbolic evaluation relation ($\downarrow_s$) for $\mathcal{F}_{\mathrm{DH}}$

$$(\textsc{Dec}_{\mathsf{S}})\ \ \mathsf{dec}^{\mathsf{sk}}_{\eta}(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} x_1\,\theta \qquad \theta = \mathrm{mgu}(\zeta = \{x_1\}_{x_2}, \eta = x_2)$$

$$(\textsc{Mult}_{\mathsf{S}})\ \ \mathsf{mult}(\zeta_1,\zeta_2) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} (x_{i_1} \times \cdots \times x_{i_n})\,\theta \qquad \begin{cases} 1 \le k < n \le l, \\ \theta = \mathrm{mgu}(\zeta_1 = x_1 \times \cdots \times x_k, \\ \zeta_2 = x_{k+1} \times \cdots \times x_n) \end{cases}$$

$$(\textsc{Inv1}_{\mathsf{S}})\ \ \mathsf{inv}(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} (\mathsf{inv}'(x_{i_1}) \times \cdots \times \mathsf{inv}'(x_{i_n}))\,\theta \quad \begin{cases} 1 \le n \le l, \\ \theta = \mathrm{mgu}(\zeta = x_1 \times \cdots \times x_n) \end{cases}$$

$$(\textsc{Inv2}_{\mathsf{S}})\ \ \mathsf{inv}'(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} x_1\,\theta \qquad \theta = \mathrm{mgu}(\zeta, {x_1}^{-1})$$

$$(\textsc{Inv3}_{\mathsf{S}})\ \ \mathsf{inv}'(\zeta) \overset{\varepsilon}{\rightsquigarrow}_{\mathsf{S}} \zeta^{-1} \qquad (\textsc{Inv4}_{\mathsf{S}})\ \ \mathsf{inv}'(\zeta) \times \eta \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} \mathsf{unit} \qquad \theta = \mathrm{mgu}(\zeta, \eta)$$

$$(\textsc{Unit1}_{\mathsf{S}})\ \ \mathsf{unit} \times \zeta \overset{\varepsilon}{\rightsquigarrow}_{\mathsf{S}} \zeta \qquad (\textsc{Unit2}_{S})\ \ \mathsf{unit} \overset{\varepsilon}{\rightsquigarrow}_{\mathsf{S}} 1$$

$$(\textsc{Exp1}_{S})\ \ \mathsf{exp}(x,\zeta) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} \mathsf{exp}(\alpha, \mathsf{mult}(x_1,\zeta)) \qquad \theta = [^{\mathsf{exp}(\alpha, x_1)}/_{x}]$$

$$(\textsc{Exp2}_{S})\ \ \mathsf{exp}(\mathsf{exp}(\xi,\eta),\zeta) \overset{\varepsilon}{\rightsquigarrow}_{\mathsf{S}} \mathsf{exp}(\xi, \mathsf{mult}(\eta,\zeta))$$

$$(\textsc{Root1}_{S})\ \ \mathsf{root}(x,\zeta) \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} \mathsf{root}(x,\zeta)\theta \qquad \theta = [^{\mathsf{exp}(\alpha, x_1)}/_{x}]$$

$$(\textsc{Root2}_{S})\ \ \mathsf{root}(\mathsf{exp}(\xi,\eta),\zeta) \overset{\varepsilon}{\rightsquigarrow}_{\mathsf{S}} \mathsf{exp}(\xi, \mathsf{mult}(\eta, \mathsf{inv}(\zeta)))$$

$$(\textsc{Ctx}_{S})\ \ \frac{\zeta \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} \zeta'}{C[\zeta] \overset{\theta}{\rightsquigarrow}_{\mathsf{S}} C\theta[\zeta']}$$

Symbolic Evaluation $\zeta \downarrow_\theta \eta$ iff $\zeta \overset{\theta_1}{\rightsquigarrow}_{\mathsf{S}} \cdots \overset{\theta_n}{\rightsquigarrow}_{\mathsf{S}} \eta$ and $\theta = \theta_1 \cdots \theta_n$

Variables $x_1, \ldots, x_n$ are fresh.

notational shorthands.

$$A \triangleq \overline{\mathsf{a1}}\langle \mathsf{exp}(\alpha, n_A)\rangle.\, \mathsf{a2}(x).\, \mathsf{let}\, z{=}\mathsf{exp}(x, n_A)\mathsf{in}\, \overline{\mathsf{a3}}\langle\{d\}_z\rangle.\, 0,$$

$$B \triangleq \mathsf{b1}(y).\, \overline{\mathsf{b2}}\langle \mathsf{exp}(\alpha, n_B)\rangle.\, \mathsf{let}\, w{=}\mathsf{exp}(y, n_B)\mathsf{in}\, \mathsf{b3}(t).\, \mathsf{let}\, t'{=}\mathsf{dec}_w(t)\mathsf{in}\, 0,$$

$$P \triangleq A \,||\, B.$$

The Diffie–Hellman protocol is subject to secrecy attacks from active adversaries. In terms of our model, discovering an attack of this type to the protocol amounts to finding a

ground trace s such that $\mathcal{C}_{\mathrm{DH}} = \langle \varepsilon,\ P \,||\, \mathsf{g}\langle d\rangle\rangle \searrow s$ and $s \not\models \mathit{Secret}(d) \stackrel{\triangle}{=} \bot \hookleftarrow \mathsf{g}\langle d\rangle$. And, indeed, such an s exists and it is as follows:

$$\overline{\mathsf{a1}}\langle \mathsf{exp}\,(\alpha, n_A)\rangle \cdot \mathsf{a2}\langle \mathsf{exp}\,(\alpha, \underline{n}_I)\rangle \cdot \overline{\mathsf{a3}}\langle \{d\}_{\mathsf{exp}\,(\alpha, \underline{n}_I \times n_A)}\rangle \cdot \mathsf{g}\langle d\rangle,$$

where $\underline{n}_I$ is any environmental name.

Intuitively, the above trace corresponds to an attack in which the environment intercepts $\mathsf{exp}\,(\alpha, n_A)$, generates a name $\underline{n}_I$ and handles $\mathsf{exp}\,(\alpha, \underline{n}_I)$ to A, who believes the message is from B. Then A computes $k = \mathsf{exp}\,(\alpha, \underline{n}_I \times n_A)$ and erroneously concludes k is a shared key known by A and B. Finally, A sends over the network the secret datum d encrypted under k, so d is revealed to the environment, that can compute k as $\mathsf{exp}\,(\mathsf{exp}\,(\alpha, n_A), \underline{n}_I)$.

We show how the above attack is detected by the verification method. First, consider the following symbolic execution starting from $\mathcal{C}_{\mathrm{DH}}$:

$$\begin{aligned}
\mathcal{C}_{\mathrm{DH}} \ &\longrightarrow_{\mathrm{S}}\ \longrightarrow_{\mathrm{S}}\ \left\langle \overline{\mathsf{a1}}\langle \mathsf{exp}\,(\alpha, n_A)\rangle \cdot \mathsf{a2}\langle x\rangle,\ \left(\mathsf{let}\, z = \mathsf{exp}\,(x, n_A)\, \mathsf{in}\, \overline{\mathsf{a3}}\langle \{d\}_z\rangle.\, 0 \parallel B \parallel \mathsf{g}(t).\, 0\right)\right\rangle_{\mathrm{S}} \\
&\stackrel{\theta_0}{\longrightarrow}_{\mathrm{S}} \left\langle \overline{\mathsf{a1}}\langle \mathsf{exp}\,(\alpha, n_A)\rangle \cdot \mathsf{a2}\langle \mathsf{exp}\,(\alpha, x')\rangle,\ \left(\overline{\mathsf{a3}}\langle \{d\}_{\mathsf{exp}\,(\alpha, x' \times n_A)}\rangle.\, 0 \parallel B\theta_0 \parallel \mathsf{g}(t).\, 0\right)\right\rangle_{\mathrm{S}} \qquad (*) \\
&\longrightarrow_{\mathrm{S}} \left\langle \overline{\mathsf{a1}}\langle \mathsf{exp}\,(\alpha, n_A)\rangle \cdot \mathsf{a2}\langle \mathsf{exp}\,(\alpha, x')\rangle \cdot \overline{\mathsf{a3}}\langle \{d\}_{\mathsf{exp}\,(\alpha, x' \times n_A)}\rangle,\ \left(B\theta_0 \parallel \mathsf{g}(t).\, 0\right)\right\rangle_{\mathrm{S}} \\
&\longrightarrow_{\mathrm{S}} \langle \overline{\mathsf{a1}}\langle \mathsf{exp}\,(\alpha, n_A)\rangle \cdot \mathsf{a2}\langle \mathsf{exp}\,(\alpha, x')\rangle \cdot \overline{\mathsf{a3}}\langle \{d\}_{\mathsf{exp}\,(\alpha, x' \times n_A)}\rangle \cdot \mathsf{g}\langle t\rangle,\ B\theta_0\rangle_{\mathrm{S}} \\
&\stackrel{\triangle}{=} \langle \sigma,\ B\theta_0\rangle_{\mathrm{S}}.
\end{aligned}$$

In step $(*)$, rule (LET_S) is applied, with $\mathsf{exp}\,(x, n_A) \downarrow_{\theta_0} \mathsf{exp}\,(\alpha, x' \times n_A)$ and $\theta_0 = [^{\mathsf{exp}\,(\alpha, x')}/_x]$, for a fixed fresh x'.

Now, we show step by step how the attack is detected by the verification method:

1. The symbolic model $\mathbf{Mod}_{\mathcal{C}}$ is computed (in practice, symbolic traces would be generated 'on-the-fly').
2. The symbolic trace σ defined above is considered.

3,4. Action $\gamma = \mathsf{g}\langle t\rangle$ is found such that γ unifies with $\beta = \mathsf{g}\langle d\rangle$, via $\theta = [^d/_t]$.

5. The set $\mathbf{SF}(\sigma\theta) = \{\sigma'\}$ is computed, where $\sigma' = \sigma\theta\theta'$, and $\theta' = [^{\hat{x}'}/_{x'}]$. As stated by Theorem 3, σ' is a consistent trace. Note, in particular, that if we let $\sigma' = \sigma'' \cdot \mathsf{g}\langle d\rangle$, then $\sigma'' \vdash d$. Indeed, there exists $\zeta = \mathsf{dec}_\xi(\{d\}_\eta) \in \mathcal{H}(\sigma'')$, with $\eta = \mathsf{exp}\,(\alpha, \hat{x}' \times n_A)$, $\xi = \mathsf{exp}\,(\mathsf{exp}\,(\alpha, n_A), \hat{x}')$, and $\zeta \rightsquigarrow_{\mathrm{S}} \rightsquigarrow_{\mathrm{S}} \mathsf{dec}_\eta(\{d\}_\eta) \rightsquigarrow_{\mathrm{S}} d$.
6. Action $\bot$ does not appear in σ', hence,
7. $(\mathsf{No},\ \sigma')$ is returned.

Note that, as indicated by Theorem 5, the concrete trace s corresponding to the attack can be recovered from σ' by mapping $\hat{x}'$ to $\underline{n}_I$.

7. An implementation: STA

Symbolic Trace Analyzer (STA) [41] is a prototype tool, written in ML, that implements some of the verification techniques described in the previous sections. Currently, STA supports shared-key, public-key cryptography and hashing, while modular exponentiation has not yet been integrated in the tool.

We now illustrate the use of STA on the Needham–Schroeder protocol introduced in Example 2, and check the authentication property *AuthAtoB* described in 3. When required to check whether *NS* satisfies *AuthAtoB*, STA finds a trace of *NS* that violates the property. The trace is reported below:

$$\overline{\mathsf{disclose}}\langle kI, kA^+, kB^+, \mathtt{id}_A, \mathtt{id}_B, \mathtt{id}_I\rangle \cdot \overline{\mathsf{a'1}}\langle\{\!|n'A, \mathtt{id}_A|\!\}_{kI^+}\rangle,$$
$$\mathsf{b1}\langle\{\!|n'A, \mathtt{id}_A|\!\}_{kB^+}\rangle \cdot \overline{\mathsf{b2}}\langle\{\!|n'A, nB|\!\}_{kA^+}\rangle \cdot \mathsf{a'2}\langle\{\!|n'A, nB|\!\}_{kA^+}\rangle,$$
$$\overline{\mathsf{a'3}}\langle\{\!|nB|\!\}_{kI^+}\rangle \cdot \mathsf{b3}\langle\{\!|nB|\!\}_{kB^+}\rangle.$$

This trace corresponds to the attack discovered by Lowe, that can be informally explained as follows. A runs two parallel sessions, one seemingly with I, and the other seemingly with B. In fact, both A and B are talking to the adversary, who intercepts all messages. This allows the adversary to re-use the nonce $n'A$ (issued by A in its interaction with I) when impersonating the role of A talking to B. Then, the adversary can induce A to decrypt message $\{\!|n'A, nB|\!\}_{kA^+}$, thus getting nB (actions $\mathsf{a'2}$ and $\mathsf{a'3}$).

This attack was found after examining 26 symbolic configurations, which took a fraction of a second on a PC with a Pentium III processor and a 64 M RAM. After repairing the flaw as suggested by Lowe, that is by inserting explicit identities inside each encrypted message, STA finds no additional attack. The exploration reached all the 60 configurations that constitute the complete symbolic state-space of the protocol, and this took again a fraction of a second. We also tried a configuration with two initiators (A, D) and two responders (B, C), where each initiator can non-deterministically choose to engage in a run with either B, C or I. STA found no attacks on this version either. The state-space consisting of 24,655 symbolic configurations was completely explored in less than one minute. It is worthwhile to notice that memory occupation is not a concern in STA, because a depth-first strategy is adopted when exploring the symbolic model on the fly.

8. Conclusions

We have proposed a framework for the analysis of security protocols and provided some sufficient conditions under which verification can be effectively performed via a symbolic method. In contrast to finite-state model checking, our method can analyse the whole infinite state space generated by a bounded number of participants. Compared to other symbolic techniques, we offer a simple and general methodology together with a regularity condition, which can be instantiated to complete verification methods for specific crypto-systems. Our method is efficient in practice, because the symbolic model is compact, and the refinement procedure at its heart is only invoked on demand and on single symbolic traces. Note that general claims on efficiency should be taken with some care, given that the protocol analysis problem is NP-hard even under very mild hypotheses (see e.g. [38]).

Appendix A. Concrete vs. symbolic semantics

We prove Theorem 1 that establishes a correspondence between the concrete and the symbolic transition systems. The proof is based on the lemma below.

Lemma A.1. *Let $\mathcal{F}^s$ be a symbolic frame, $\mathcal{C}$ be a symbolic configuration and $\mathcal{C}\rho$ be a configuration. Then*:

(1) *$\mathcal{C}\rho \longrightarrow \mathcal{C}'$ implies that there exist $\mathcal{C}_1$, θ and ρ_0 such that $\rho = (\theta\rho_0)_{|\mathrm{dom}(\rho)}$, $\mathcal{C} \stackrel{\theta}{\longrightarrow}_{\mathrm{S}} \mathcal{C}_1$ and $\mathcal{C}' = \mathcal{C}_1\rho_0$.*

(2) *$\mathcal{C} \stackrel{\theta}{\longrightarrow}_{\mathrm{S}} \mathcal{C}_1$ and $\rho = \theta\rho_0$, for some ρ_0 such that $\mathcal{C}_1\rho_0$ is a configuration, imply that $\mathcal{C}\rho \longrightarrow \mathcal{C}_1\rho_0$.*

Proof. (1) By induction on the rules of the transition relation $\longrightarrow$. The cases (INP) and (PAR) are trivial.

(OUT) $\langle \sigma\rho,\ \overline{\mathsf{a}}\langle\zeta\rho\rangle.\, A\rho\rangle \longrightarrow \langle \sigma\rho{\cdot}\overline{\mathsf{a}}\langle M\rangle,\ A\rho\rangle$, and $\zeta\rho \downarrow M$. By def. of symbolic frame, $\zeta \downarrow_\theta N, \rho{=}(\theta\rho_0)_{|\mathrm{dom}(\rho)}$ and $M{=}N\rho_0$. Then, $\langle \sigma,\ \overline{\mathsf{a}}\langle\zeta\rangle.\, A\rangle_{\mathrm{S}} \stackrel{\theta}{\longrightarrow}_{\mathrm{S}} \langle \sigma\theta{\cdot}\overline{\mathsf{a}}\langle N\rangle,\ A\theta\rangle_{\mathrm{S}}$ and also $\langle \sigma\theta{\cdot}\overline{\mathsf{a}}\langle N\rangle,\ A\theta\rangle_{\mathrm{S}}\rho_0 = \langle \sigma\rho{\cdot}\overline{\mathsf{a}}\langle M\rangle,\ A\rho\rangle$. Note that above we have exploited that $\mathrm{v}(A) \subseteq \mathrm{dom}(\rho)$ and $\mathrm{v}(\sigma) \subseteq \mathrm{dom}(\rho)$.

(LET) $\langle \sigma\rho,\ \mathsf{let}\ y{=}\zeta\rho\,\mathsf{in}\,A\rho\rangle \longrightarrow \langle \sigma\rho,\ A\rho[\eta/y]\rangle$, with $\zeta\rho \downarrow \eta$. By def. of symbolic frame, $\zeta \downarrow_\theta \xi$, $\rho = (\theta\rho_0)_{|\mathrm{dom}(\rho)}$ and $\eta = \xi\rho_0$. Then, $\langle \sigma | let\ y = \zeta\ \mathsf{in}\ A\rangle_{\mathrm{S}} \stackrel{\theta}{\longrightarrow}_{\mathrm{S}} \langle \sigma\theta,\ A\theta[\xi/y]\rangle_{\mathrm{S}}$ where $\langle \sigma\theta,\ A\theta[\xi/y]\rangle_{\mathrm{S}}\rho_0 = \langle \sigma\rho,\ A\rho[\eta/y]\rangle$.

(MATCH) $\langle \sigma\rho,\ [\zeta\rho = \eta\rho]A\rho\rangle \longrightarrow \langle \sigma\rho,\ A\rho\rangle$, where $\zeta\rho \downarrow \xi$, $\eta\rho \downarrow \xi$. By definition of symbolic frame, $\zeta\rho \downarrow \xi$ implies $\zeta \downarrow_{\theta_1} \xi_1$, where $\rho = (\theta_1\rho_0{}')_{|\mathrm{dom}(\rho)}$ and $\xi = \xi_1\rho_0{}'$, for some ρ_0'. Also $\eta\rho = \eta\theta_1\rho_0{}'$ ($\mathrm{v}(\eta) \subseteq \mathrm{dom}(\rho)$), and, thus, $\eta\theta_1 \downarrow_{\theta_2} \xi_2$, where $\rho_0{}' = (\theta_2\rho_0{}'')_{|\mathrm{dom}(\rho_0')}$ and $\xi_2\rho_0{}'' = \xi$, for some $\rho_0{}''$. Since $\xi = \xi_1\rho_0{}' = \xi_1\theta_2\rho_0{}'' = \xi_2\rho_0''$, then $\xi_1\theta_2$ and ξ_2 are unifiable. Let $\theta_3 = \mathrm{mgu}(\xi_1\theta_2, \xi_2)$, i.e. $\rho_0{}'' = \theta_3\rho_0$ for some ρ_0. We let $\theta \stackrel{\triangle}{=} \theta_1\theta_2\theta_3$. Then, $\rho = \theta\rho_0$ and $\langle \sigma,\ [\zeta = \eta]A\rangle_{\mathrm{S}} \stackrel{\theta}{\longrightarrow}_{\mathrm{S}} \langle \sigma\theta,\ A\theta\rangle_{\mathrm{S}}$, where $\langle \sigma\theta,\ A\theta\rangle_{\mathrm{S}}\rho_0 = \langle \sigma\rho,\ A\rho\rangle$.

(2) The proof is by induction on $\longrightarrow_{\mathrm{S}}$. It is similar to the previous case and then it is omitted.

Corollary A.1 (*Theorem 1*). *Let $\mathcal{F}^s$ be a symbolic frame, $\mathcal{C}$ be an initial configuration and s be a trace of $\mathcal{F}$. Then $\mathcal{C} \searrow s$ if and only if there exists σ such that $\mathcal{C} \searrow_{\mathrm{S}} \sigma$ and s is a solution of σ.*

Proof. The proof easily follows by a routine induction on the number n of execution steps, using Lemma A.1. In particular, for the 'if' direction, we exploit part (2) and note that $\mathcal{C}\rho = \mathcal{C}$, since $\mathcal{C}$ is ground. □

Appendix B. Proof of Lemma 1

The proof of Lemma 1 relies on Lemma B.1 below. The latter essentially states that any input message that violates the 'sf-ness' of a consistent symbolic trace can be decomposed so to satisfy the premises of either of the refinement rules.

Lemma B.1 (*Context lemma*). *Let $\mathcal{F}^s$ be a regular frame and σ be a marked symbolic trace in sf. If ρ satisfies $\sigma \cdot \mathsf{a}\langle M\rangle$ and $\sigma \not\vdash M$, then either*:
(a) $\exists C[\cdot], N$ s.t. $N \notin \mathbf{b}(\sigma) \cup \mathcal{V} \cup \hat{\mathcal{V}}$, $M = C[N]$, *and* $N\rho \in \mathbf{b}(\sigma)\rho$ *or*
(b) $\exists x \in \mathrm{v}(M)$ s.t. $\sigma\rho \vdash \rho(x)$.

Proof. Since ρ satisfies $\sigma \cdot \mathsf{a}\langle M\rangle$, then $\sigma\rho \vdash M\rho$ and $\forall \hat{x} \in \mathrm{v}(M)$, $\sigma\rho \vdash \rho(\hat{x})$. Since $\mathcal{F}$ is regular, $M\rho \in \mathcal{H}(\mathbf{b}(\sigma\rho))$. The proof is by induction on the least index i such that $M\rho \in \mathcal{H}^i(\mathbf{b}(\sigma\rho))$.

($i = 0$) $M\rho \in \mathbf{b}(\sigma\rho)$. Depending on the form of M, there are three cases:
- $M = \hat{x}$ or $M = \underline{a}$. These cases cannot arise, as they would imply $\sigma \vdash M$.
- $M = x$. Then, (b) holds.
- $M \notin \mathcal{V} \cup \hat{\mathcal{V}}$. $M \notin \mathbf{b}(\sigma)$ as, otherwise, it would follow $\sigma \vdash M$. Since $\mathcal{F}$ is regular, $\mathbf{b}(\sigma\rho) \subseteq \mathbf{b}(\sigma)\rho$, thus, there exists $N' \in \mathbf{b}(\sigma)$ such that $M\rho = N'\rho$. Then, we define $C \triangleq [\cdot]$, $N \triangleq M$. $N\rho \in \mathbf{b}(\sigma)\rho$, as $N' \in \mathbf{b}(\sigma)$ and $N\rho = N'\rho$; thus, (a) holds.

($i > 0$) $M = f(\widetilde{M}')$, where $\widetilde{M}'\rho \subseteq \mathcal{H}^{i-1}(\sigma\rho)$. Note that, for some $M_i{}' \in \widetilde{M}'$, $\sigma \not\vdash M_i{}'$, as otherwise $\sigma \vdash M$, by definition of $\mathcal{H}(\cdot)$. By induction hypothesis, either (aa) there exist C' and N' s.t. $N' \notin \mathbf{b}(\sigma) \cup \mathcal{V} \cup \hat{\mathcal{V}}$, $M_i{}' = C'[N']$, and $N'\rho \in \mathbf{b}(\sigma)\rho$ or (bb) there exists $x \in \mathrm{v}(M_i{}')$ such that $\sigma\rho \vdash \rho(x)$. Obviously (bb) implies (b). If (aa) holds, then we choose $N \triangleq N'$ and $C[\cdot] \triangleq f(\widetilde{M}'_1, C'[\cdot], \widetilde{M}'_2)$, where $\widetilde{M}' = (\widetilde{M}'_1, M_i{}', \widetilde{M}'_2)$, thus (a) holds true. □

Lemma B.2 (*Lemma 1*). *Let $\mathcal{F}^s$ be a regular frame and σ be a marked symbolic trace in $\mathcal{F}^s$. Suppose that ρ satisfies σ. Then either σ is in sf, or there are σ' and ρ' such that $\sigma \succ \sigma'$, $\sigma\rho = \sigma'\rho'$, and ρ' satisfies σ'.*

Proof. Suppose that σ is *not* in sf and let σ_1 be the longest prefix of σ which is in sf. This means that $\sigma = \sigma_1 \cdot \mathsf{a}\langle M\rangle \cdot \sigma_2$, for some M s.t. $\sigma_1 \not\vdash M$. Since ρ satisfies σ, we must have $\sigma_1\rho \vdash M\rho$. By Lemma B.1, either (a) there exist $C[\cdot], N, N'$ such that $N \notin \mathbf{b}(\sigma_1) \cup \mathcal{V} \cup \hat{\mathcal{V}}$, $M = C[N]$, $N' \in \mathbf{b}(\sigma_1)$ and $N'\rho = N\rho$, or (b) there exists $x \in \mathrm{v}(M)$ such that $\sigma_1\rho \vdash \rho(x)$.

If (a) is the case, there exists $\theta = \mathrm{mgu}(N, N')$ and ρ is an instance of θ as a substitution. By rule (REF$_1$), $\sigma \succ \sigma' = \sigma\theta[\tilde{y}/\tilde{y}]$, for an appropriate renaming $[\tilde{y}/\tilde{y}]$. Furthermore, $\rho = \theta[\tilde{y}/\tilde{y}]\rho'$, for some ground substitution ρ'. Also note that, thanks to the renaming $[\tilde{y}/\tilde{y}]$, for each $\hat{x} \in \mathrm{v}(\sigma')$, $\sigma'\backslash\hat{x}$ is not longer than $\sigma\backslash\hat{x}$, and this guarantees that ρ' satisfies σ'. Thus we have found σ' and ρ' as required by the statement of the lemma.

If (b) holds, then we can apply (REF$_2$), define $\sigma' = \sigma[\hat{x}/x]$ and $\rho' = [x/\hat{x}]\rho$, and the thesis will follow. □

Appendix C. Proofs on $\mathcal{F}^s_{pk}$

C.1. $\mathcal{F}^s_{pk}$ is a Symbolic Frame

Notation: By $\zeta \overset{\theta}{\rightsquigarrow}_V \eta$ we mean that $\zeta \overset{\theta}{\rightsquigarrow}_{\mathrm{s}} \eta$, with $(\mathrm{v}(\theta) \setminus \mathrm{v}(\zeta)) \subseteq V$, and by $\zeta \overset{\theta}{\rightsquigarrow}{}^*_V \eta$ that $\zeta \overset{\theta_1}{\rightsquigarrow}_{V_1} \cdots \overset{\theta_n}{\rightsquigarrow}_{V_n}$, with $V = \bigcup_1^n V_i$, for some $n \geq 0$ and $\theta = \theta_1 \cdots \theta_n$. Moreover, by $\theta \setminus V$ we mean $\theta \setminus (V \times \mathcal{E}_\Sigma)$.

Lemma C.1. *Suppose* $\rightsquigarrow$ *and* $\rightsquigarrow_{\mathrm{S}}$ *are defined, respectively, as in Tables* 1 *and* 4. *Then*:

(1) *If* $\zeta\rho\rightsquigarrow\eta$ *then* $\forall X \supseteq \mathrm{dom}(\rho)$, $\exists\eta', \rho_0$ *and* $V : X \cap V = \emptyset$, *such that* $\zeta \overset{\theta}{\rightsquigarrow}{}^{*}_{V} \eta'$, $\rho = (\theta\rho_0) \setminus V$ *and* $\eta'\rho_0 = \eta$. *Furthermore,* $\eta \in \mathcal{M}$ *implies* $\eta' \in \mathcal{M}$.

(2) *If* $\zeta \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} \eta$ *and* $\rho = \theta\rho_0$, *for some* ρ_0, *then* $\zeta\rho\rightsquigarrow^{*}\eta\rho_0$.

Proof. (1) By induction on $\zeta\rho\rightsquigarrow\eta$. We only consider the most interesting case of decryption.

(DEC) Let $\zeta\rho = \mathsf{dec}^{\mathsf{pk}}_{\psi^-}(\{\!|\eta|\!\}_{\psi^+})$. Then, $\zeta = \mathsf{dec}^{\mathsf{pk}}_{\zeta''}(\zeta')$, with $\zeta'\rho = \{\!|\eta|\!\}_{\psi^+}$ and $\zeta''\rho = \psi^-$. Let $V = \{x_1, x_2\}$ be such that $V \cap X = \emptyset$. Then, there exists $\theta' = \mathrm{mgu}(\zeta' = \{\!|x_1|\!\}_{x_2^+}, \zeta'' = x_2^-)$ and $\rho = (\theta'\rho'_0) \setminus V$, for some ρ'_0. By definition of $\rightsquigarrow_{\mathrm{S}}$, $\zeta \overset{\theta'}{\rightsquigarrow}_{V} x_1\theta' \triangleq \eta''$, with $\eta''\rho'_0 = x_1\theta'\rho'_0 = \zeta\theta'\rho'_0 = \eta$. Now let $\eta = M \in \mathcal{M}$. Then $\eta''\rho'_0 = M$ implies $\rho'_0 = [\widetilde{x}^+/\widetilde{x}]\rho_0$ (where $\widetilde{x}$ are the variables which occur in key position in η'') for some ρ_0, and $\eta'' \overset{\theta}{\rightsquigarrow}{}^{*}_{\emptyset} \eta''\theta'' \triangleq M' \in \mathcal{M}$, with $\theta'' \triangleq [\widetilde{x}^+/\widetilde{x}]$. Finally, $\zeta \overset{\theta}{\rightsquigarrow}{}^{*}_{V} \eta'$, with $\theta \triangleq \theta'\theta''$ and $\eta'\rho_0 = \eta''\theta''\rho_0 = \eta''\rho'_0 = \eta \in \mathcal{M}$.

(2) By structural induction on the rules $\rightsquigarrow_{\mathrm{S}}$.

(ENC$_S$) $\{\!|\zeta|\!\}_x \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} \{\!|\zeta\theta|\!\}_{x^+}$, with $\theta = [x^+/x]$. We prove that $\{\!|\zeta\rho|\!\}_{x\rho}\rightsquigarrow^{*} \{\!|\zeta\theta|\!\}_{x^+}\rho_0$ in 0 steps. Indeed, $\{\!|\zeta\theta|\!\}_{x^+}\rho_0 = \{\!|\zeta\theta|\!\}_{\theta(x)}\rho_0 = \{\!|\zeta\rho|\!\}_{\rho(x)}$.

(PRJ$_S$) $\pi_i(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} x_i\theta$,] with $\theta = \mathrm{mgu}(\zeta, \langle x_1, x_2\rangle)$. By definition of θ, $\pi_i(\zeta\rho) = \pi_i(\zeta\theta\rho_0) = \pi_i(\langle x_1\theta\rho_0, x_2\theta\rho_0\rangle)$ and $\pi_i(\langle x_1\theta\rho_0, x_2\theta\rho_0\rangle)\rightsquigarrow x_i\theta\rho_0$, by applying rules of $\rightsquigarrow$.

(DEC$_S$) $\mathsf{dec}^{\mathsf{pk}}_{\psi}(\zeta) \overset{\theta}{\rightsquigarrow}_{\mathrm{S}} x_1\theta$, with $\theta = \mathrm{mgu}(\zeta = \{\!|x_1|\!\}_{x_2^+}, \psi = x_2^-)$. By definition of θ, $\mathsf{dec}^{\mathsf{pk}}_{\psi\rho}(\zeta\rho) = \mathsf{dec}^{\mathsf{pk}}_{\psi\theta\rho_0}(\zeta\theta\rho_0) = \mathsf{dec}^{\mathsf{pk}}_{x_2^-\theta\rho_0}(\{\!|x_1\theta\rho_0|\!\}_{x_2^+\theta\rho_0})$. Finally, by applying rules of $\rightsquigarrow$, $\mathsf{dec}^{\mathsf{pk}}_{x_2^-\theta\rho_0}(\{\!|x_1\theta\rho_0|\!\}_{x_2^+\theta\rho_0})\rightsquigarrow x_1\theta\rho_0$.

(CTX$_S$) By induction hypothesis, $\zeta\rho\rightsquigarrow^{*}\zeta'\rho_0$ and, by rules of $\rightsquigarrow$, $C\rho[\zeta\rho]\rightsquigarrow^{*}C\rho[\zeta'\rho_0]$, i.e. $(C[\zeta])\rho\rightsquigarrow^{*}(C\theta[\zeta'])\rho_0$. □

Proposition C.1. (a) *If* $\zeta\rho\rightsquigarrow^{*}\eta$ *then* $\forall X \supseteq \mathrm{dom}(\rho)$, $\exists V : X \cap V = \emptyset$ *such that* $\zeta \overset{\theta}{\rightsquigarrow}{}^{*}_{V} \eta'$, *with* $\rho = (\theta\rho_0) \setminus V$ *and* $\eta'\rho_0 = \eta$. *Furthermore,* $\eta \in \mathcal{M}$ *implies* $\eta' \in \mathcal{M}$.

(b) *If* $\zeta \overset{\theta}{\rightsquigarrow}{}^{*}_{\mathrm{S}} \eta$ *and* $\rho = \theta\rho_0$, *for some* ρ_0, *then* $\zeta\rho\rightsquigarrow^{*}\eta\rho_0$.

Proof. (a) By induction on the number n of steps of relation $\rightsquigarrow$, such that $\zeta\rho\rightsquigarrow^{n}\eta$. The case $n = 0$ is trivial. Suppose $n > 0$ and $\zeta\rho\rightsquigarrow^{n-1}\xi\rightsquigarrow\eta$. By induction hypothesis, $\zeta \overset{\theta'}{\rightsquigarrow}{}^{*}_{V'} \xi'$, with $V' \cap X = \emptyset$, $\rho = (\theta'\rho'_0) \setminus V'$ and $\xi'\rho'_0 = \xi$. By $\xi = \xi'\rho'_0\rightsquigarrow\eta$ and by Lemma C.1 (applied to $X \cup V'$) it follows that $\xi' \overset{\theta''}{\rightsquigarrow}{}^{*}_{V''} \eta'$, with $V'' \cap (X \cup V') = \emptyset$, $\rho'_0 = (\theta''\rho_0) \setminus V''$ and $\eta'\rho_0 = \eta$. Now, let $\theta = \theta'\theta''$ and $V = V' \cup V''$. Then, $\zeta \overset{\theta}{\rightsquigarrow}{}^{*}_{V} \eta$, with $V \cap X = \emptyset$ and $\rho = (\theta'\rho'_0) \setminus V' = (\theta'((\theta''\rho_0) \setminus V'')) \setminus V' = ((\theta'\theta''\rho_0) \setminus V'') \setminus V' = (\theta\rho_0) \setminus V$, exploiting the fact that $\mathrm{v}(\theta') \subseteq (X \cup V')$. Furthermore, if $\eta \in \mathcal{M}$, then $\eta' \in \mathcal{M}$, by Lemma C.1.

(b) By induction on the number n of steps of relation $\rightsquigarrow_{\mathrm{S}}$, i.e. such that $\zeta \overset{\theta_1}{\rightsquigarrow_{\mathrm{S}}} \zeta_1 \overset{\theta_2}{\rightsquigarrow_{\mathrm{S}}} \cdots \overset{\theta_n}{\rightsquigarrow_{\mathrm{S}}} \eta$, with $\rho = \theta_1\theta_2\cdots\theta_n\rho_0$, for some ρ_0. The case $n = 0$ is trivial. Suppose $\zeta \overset{\theta_1}{\rightsquigarrow_{\mathrm{S}}} \zeta_1 \overset{\theta_2}{\rightsquigarrow_{\mathrm{S}}} \cdots \overset{\theta_n}{\rightsquigarrow_{\mathrm{S}}} \zeta_n \overset{\theta_{n+1}}{\rightsquigarrow_{\mathrm{S}}} \eta$. By induction hypothesis, $\zeta\rho \downarrow \zeta_n\rho_0$, for some ρ_0 such that $\rho = \theta_1\cdots\theta_n\rho_0$. By applying Lemma C.1 to $\zeta_n\rho_0$, it follows that $\zeta\rho \rightsquigarrow^* \zeta_n\rho_0 \rightsquigarrow \eta\rho_0'$, for some ρ_0'. □

Corollary C.1 (*Proposition 1*). *$\mathcal{F}^s_{pk}$ is a symbolic frame.*

Proof. It trivially follows by Proposition C.1 with $X \triangleq \mathrm{dom}(\rho)$. □

C.2. $\mathcal{F}^s_{pk}$ is a regular frame

We now prove that $\mathcal{F}^s_{pk}$ is a regular frame. Propositions C.2 and C.3 below state that $\mathbf{b}_{pk}$ satisfies, respectively, Conditions 1 and 2 in the definition of regular frame (Definition 10). In particular, note that Proposition C.2 generalises Condition 1 in Definition 10, since σ below can be either a trace or a solved form.

Proposition C.2. *Let σ be a sf or a trace in $\mathcal{F}^s_{pk}$. Suppose* $\mathrm{v}(M) \subseteq \widehat{\mathcal{V}}$. *Then, $\sigma \vdash M$ iff $M \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$.*

Proof. Suppose $\sigma \vdash M$. By induction on the structure of M.
- $M \in \mathcal{EN}$. Then, $M \in \mathcal{H}^0(\mathbf{b}_{pk}(\sigma))$.
- $M = (\hat{x})^{\pm}$. Then, $M \in \mathcal{H}^1(\mathbf{b}_{pk}(\sigma))$.
- $M = (m)^{\pm}$. Then, $(m)^{\pm} \in \mathcal{H}^1(\mathbf{b}_{pk}(\sigma))$, by definition.
- $M = \langle M_1, M_2\rangle$. By induction hypothesis $M_1,\ M_2 \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$ and, consequently, $M \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$.
- $M = \{\!|M'|\!\}_{u^+}$. If $\sigma \nvdash \langle M', u^+\rangle$, $M \in \mathbf{b}_{pk}(\sigma)$, by definition. Otherwise, by induction hypothesis, $u^+,\ M' \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$ and, thus, $\{\!|M'|\!\}_{u^+} = M \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$.

On the other hand, suppose $M \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$. By induction on the least j such that $M \in \mathcal{H}^j(\mathbf{b}_{pk}(\sigma))$.

($j = 0$) There are two cases. If $M \in \widehat{\mathcal{V}} \cup \mathcal{EN}$, trivially $\sigma \vdash M$. Else $M \in \mathbf{b}_{pk}(\sigma)$ and it follows by definition that $\sigma \vdash M$.

($j > 0$) Suppose $M = \{\!|M'|\!\}_{u^+}$ (the case $M = \langle M_1, M_2\rangle$ is analogous), with $M',\ u^+ \in \mathcal{H}^{j-1}(\mathbf{b}_{pk}(\sigma))$. Then, the thesis follows by induction hypothesis on M' and u^+.

□

The proof of Proposition C.3 relies on the lemmata below. Lemma C.2 says that the deducibility relation on messages, $\sigma \vdash M$, is preserved by ground substitutions ρ, under suitable conditions. Lemma C.3 generalises Proposition C.2 to arbitrary terms ζ. Lemma C.4 is a sort of 'converse' of Lemma C.2 (i.e., from $\sigma\rho \vdash \zeta\rho$ it is deduced that $\sigma \vdash \zeta$, under appropriate conditions).

Lemma C.2. *Let σ be in sf and ρ be a substitution that satisfies σ. If $\sigma \vdash M$, then*:
(1) $\mathrm{v}(M) \subseteq \widehat{\mathcal{V}}$.
(2) $\sigma\rho \vdash M\rho$.

Proof. Let σ_0 be the shortest prefix of σ such that $\sigma_0 \vdash M$. The proof is by induction on $|\sigma_0|$. The case $|\sigma_0| = 0$ is obvious. If $|\sigma_0| > 0$, we proceed by induction on the least index j such that there exists $\eta \in \mathcal{H}^j(\sigma_0)$, with $\eta \downarrow M$.

($j = 0$) Necessarily $\eta = M$. The case $M \in \widehat{\mathcal{V}} \cup \mathcal{EN}$ is obvious. If $M \in \sigma_0$ then $M\rho \in \sigma_0\rho$ and thus $\sigma_0\rho \vdash M\rho$, i.e. (2). We now prove (1). It is not the case that M is an input message, i.e. $\sigma_0 = \sigma_1 \cdot \mathsf{a}\langle M\rangle \cdot \sigma_2$, as it would imply $\sigma_1 \vdash M$ (by definition of sf), in contradiction to the hypothesis on σ_0. Then necessarily M is an output message, i.e. $\sigma_0 = \sigma_1 \cdot \overline{\mathsf{a}}\langle M\rangle \cdot \sigma_2$. By definition of symbolic trace, it follows that $\mathrm{v}(M) \cap \mathcal{V} \subseteq \mathrm{v}(\sigma_1) \cap \mathcal{V}$. By induction hypothesis on σ_1, $\mathrm{v}(\sigma_1) \cap \mathcal{V} = \emptyset$ (otherwise, (1) would be violated for σ_1) and, thus, $\mathrm{v}(M) \subseteq \widehat{\mathcal{V}}$.

($j > 0$) Suppose $\eta = \mathsf{dec}^{\mathsf{pk}}_{\zeta_2}(\zeta_1)$ with $\zeta_1 \downarrow \{\!|M|\!\}_{u^+}$ and $\zeta_2 \downarrow u^-$. By induction hypothesis, $\mathrm{v}(\{\!|M|\!\}_{u^+}, u^-) \subseteq \widehat{\mathcal{V}}$, which implies (1) for M. Also, by induction hypothesis, $\sigma_0\rho \vdash \{\!|M\rho|\!\}_{(u^+\rho)}$ and $\sigma_0\rho \vdash (u^-)\rho$. It follows that $\sigma_0\rho \vdash M\rho$. The other cases ($\eta = \zeta^+$, $\eta = \zeta^-$, $\eta = \langle\zeta_1, \zeta_2\rangle$, $\eta = \pi_i(\zeta)$ are similar). □

Remark C.1. (1) It is straightforward to prove the analogue of Lemma C.2(1) for traces, i.e.: Let s be a trace and M a message such that $s \vdash M$, then $\mathrm{v}(M) \subseteq \widehat{\mathcal{V}}$.
(2) Let σ be a sf. Then $\mathrm{v}(\sigma) \subseteq \widehat{\mathcal{V}}$. This fact trivially follows by Lemma C.2(1).

Let us now generalise the definition of deduction relation $\vdash$ to arbitrary terms, by letting $\sigma \vdash \zeta$ if and only if $\exists \eta \in \mathcal{H}(\sigma) : \eta \downarrow \zeta$.

Lemma C.3. *Let σ be in sf in $\mathcal{F}^s_{pk}$. Then, $\sigma \vdash \zeta$ if and only if $\zeta \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$.*

Proof. The 'if' part of the lemma is proved by an easy induction on the least j such that $\zeta \in \mathcal{H}^i(\mathbf{b}_{pk}(\sigma))$. Conversely, suppose $\eta \downarrow \zeta$, for some $\eta \in \mathcal{H}(\sigma)$. The proof is by induction on the least j such that $\eta \in \mathcal{H}^j(\sigma)$.

($j = 0$) Then, either $\eta = \zeta = M \in \sigma$ for some M, and the result follows from Lemma C.1, or $\zeta, \eta \in \mathcal{EN} \cup \widehat{\mathcal{V}}$, and the result is trivial.

($j > 0$) We distinguish the outermost operator of η. The only non-trivial case is $\eta = \mathsf{dec}^{\mathsf{pk}}_{\eta_1}(\eta_2)$, where $\eta_2 \downarrow \{\!|\zeta|\!\}_{\zeta'^+}$ and $\eta_1 \downarrow \zeta'^-$. By induction on j, $\{\!|\zeta|\!\}_{\zeta'^+} \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$. The are two cases:

1. $\{\!|\zeta|\!\}_{\zeta'^+} \in \mathcal{H}^0(\mathbf{b}_{pk}(\sigma))$. Then, it must be $\{\!|\zeta|\!\}_{\zeta'^+} = \{\!|M|\!\}_{u^+} \in \mathbf{b}_{pk}(\sigma)$, hence $\zeta = M$, for some M. Again, the thesis follows from Lemma C.1.
2. $\{\!|\zeta|\!\}_{\zeta'^+} \in \mathcal{H}^i(\mathbf{b}_{pk}(\sigma))$ with $i > 0$, hence $\zeta \in \mathcal{H}(\mathbf{b}_{pk}(\sigma))$, which is the thesis for this case. □

Lemma C.4. *Let σ be in sf, ρ satisfy σ and $A = \{m, m^+, m^- \mid m \in \mathcal{N}\}$. If $\sigma\rho \vdash \eta$, with η ground, then there exists χ, with $\mathrm{v}(\chi) \subseteq \mathrm{v}(\sigma)$ such that $\sigma \vdash \chi$ and $\chi\rho = \eta$. Moreover, if $\eta \in A$ then $\chi = \eta$.*

Proof. Let σ_0 be the shortest prefix of σ such that $\sigma_0\rho \vdash \eta$. The proof is by induction on $|\sigma_0|$. If $|\sigma_0| = 0$, we take $\chi \triangleq \eta$. And, indeed, $\sigma_0 \vdash \chi$ since $\mathrm{n}(\eta) \subseteq \mathcal{EN}$. If $|\sigma_0| > 0$, we proceed by induction on the least index j such that there exists $\zeta \in \mathcal{H}^j(\sigma_0\rho)$, with $\zeta \downarrow \eta$.

($j = 0$) It must be $\eta = \zeta = M \in \sigma_0 \cup \mathcal{EN}$. The case $M \in \mathcal{EN}$ is obvious. If $M \in \sigma_0\rho$, there exists $\chi = N \in \sigma_0$ such that $N\rho = M$ and obviously $\mathrm{v}(N) \subseteq \mathrm{v}(\sigma)$. Furthermore, it is not the case that $N = (\hat{x})^{\pm}$, as this would imply $(\sigma_0 \setminus \hat{x})\rho \vdash \rho(\hat{x})^{\pm} = M$, with $\sigma_0 \setminus \hat{x}$ shorter than σ_0, contradicting the minimality of σ_0. Also, it is not the case that $N = (x)^{\pm}$, as $\mathrm{v}(\sigma_0) \subseteq \widehat{\mathcal{V}}$, by Remark C.1(2). It follows that if $M = (m)^{\pm} \in A$ then $N = M$, since $N\rho = M$.

($j > 0$) There are different cases, depending on the outermost operator of ζ. Here we consider the only non-trivial case, i.e. $\zeta = \mathsf{dec}^{\mathsf{pk}}_{\zeta_1}(\zeta_2)$, where $\zeta_2 \downarrow \{\!|\eta|\!\}_{\psi^+}$ and $\zeta_1 \downarrow \psi^-$. By induction hypothesis (internal or external) it follows that there are χ_1 and χ_2 such that:

(i) $\sigma_0 \vdash \chi_1 = \{\!|\chi|\!\}_{\psi'}$, with $\mathrm{v}(\{\!|\chi|\!\}_{\psi'}) \subseteq \mathrm{v}(\sigma_0)$ and $\{\!|\chi|\!\}_{\psi'}\rho = \{\!|\eta|\!\}_{\psi^+}$, for some χ, ψ' (note that we can assume w.l.o.g. that $\chi_1 \notin \widehat{\mathcal{V}}$, by considering the shortest prefix σ'_0 of σ_0 s.t. $\sigma'_0\rho \vdash \{\!|\eta|\!\}_{\psi^+}$).

(ii) $\sigma_0 \vdash \chi_2$, with $\mathrm{v}(\chi_2) \subseteq \mathrm{v}(\sigma_0)$ and $\chi_2\rho = \psi^-$. Moreover, if $\psi^- \in A$ then $\chi_2 = \psi^-$.

By Lemma C.3 it follows that $\{\!|\chi|\!\}_{\psi'} \in \mathcal{H}(\mathbf{b}_{pk}(\sigma_0))$, hence there are two cases:

(1) $\{\!|\chi|\!\}_{\psi'} \in \mathbf{b}_{pk}(\sigma_0)$. In this case, $\{\!|\chi|\!\}_{\psi'} = \{\!|N|\!\}_{k^+}$, for some N, k. (Note that it cannot arise that $\psi' = \hat{x}^+, \underline{a}^+$, by definition of $\mathbf{b}_{pk}(\cdot)$, or $\psi' = x^+$ since $\mathrm{v}(\psi') \subseteq \mathrm{v}(\sigma_0) \subseteq \widehat{\mathcal{V}}$ by Remark C.1 (2).) Hence, $\psi = k$, from (i).
By (ii) it follows that $\chi_2 = k^-$. By $\sigma_0 \vdash \{\!|\chi|\!\}_{k^+}$ and $\sigma_0 \vdash k^-$ it follows that $\sigma_0 \vdash \chi$, where by (i) $\chi\rho = \eta$, and $\mathrm{v}(\chi) \subseteq \mathrm{v}(\sigma_0)$. Moreover, if $\eta \in A$ then obviously $\chi = \eta$ (in particular, it is not the case that $\chi = \hat{x}$, by the minimality of σ_0).

(2) $\{\!|\chi|\!\}_{\psi'^+} \in \mathcal{H}^t(\mathbf{b}_{pk}(\sigma_0))$, $t > 0$. Thus, $\chi, \psi'^+ \in \mathcal{H}(\mathbf{b}_{pk}(\sigma_0))$. By Lemma C.3, $\sigma_0 \vdash \chi$ and by (i) $\chi\rho = \eta$ and $\mathrm{v}(\chi) \subseteq \mathrm{v}(\sigma_0)$. Finally, $\eta \in A$ implies $\chi = \eta$, by the same reasoning as in (1) (from (i)).

Proposition C.3. *Let σ be in sf in $\mathcal{F}^s_{pk}$. Then, $\mathbf{b}_{pk}(\sigma\rho) \subseteq \mathbf{b}_{pk}(\sigma)\rho$, for any ρ that satisfies σ.*

Proof. Suppose $M \in \mathbf{b}_{pk}(\sigma\rho)$. Note that, by Remark C.1(1) and by definition of $\mathbf{b}_{pk}(\cdot)$, M is necessarily ground. We have to prove that there exists $N \in \mathbf{b}_{pk}(\sigma)$ such that $N\rho = M$. Let σ_0 be the shortest prefix of σ such that $\sigma_0\rho \vdash M$. Clearly, $M \in \mathbf{b}_{pk}(\sigma_0\rho)$ too. We distinguish the two possible cases, depending on the structure of M.

- $M = (k)^{\pm}$. Then $\sigma_0 \vdash (k)^{\pm}$ by Lemma C.4 and obviously $\sigma \vdash (k)^{\pm}$. In this case, we take $N \triangleq (k)^{\pm} \in \mathbf{b}_{pk}(\sigma)$.
- $M = \{\!|M'|\!\}_{k^+}$, with $\sigma\rho \not\vdash \langle M', m^+\rangle$. By Lemma C.4 there exists χ such that $\sigma_0 \vdash \chi$, $\mathrm{v}(\chi) \subseteq \mathrm{v}(\sigma_0)$ and $\chi\rho = M$. By the hypotheses on σ_0, $\chi \neq \hat{x}$ and so it must be $\chi = \{\!|\chi'|\!\}_{\psi}$, for some χ', ψ. Now, by Lemma C.3, it follows that $\chi \in \mathcal{H}^j(\mathbf{b}_{pk}(\sigma_0))$, for some $j \geq 0$. But it holds that $\sigma \not\vdash \langle \chi', \psi\rangle$, otherwise by Lemma C.2 it would follow that $\sigma\rho \vdash \langle \chi'\rho, \psi\rho\rangle = \langle M', k^+\rangle$, contradicting the hypothesis that $M \in \mathbf{b}_{pk}(\sigma\rho)$. Therefore it must be $j = 0$, hence $\chi = N \in \mathbf{b}_{pk}(\sigma)$ and $N\rho = M$. □

Theorem C.1 (*Theorem 2*). *$\mathcal{F}^s_{pk}$ is a regular frame.*

Proof. The two conditions of regularity follow, respectively, from Proposition C.2 (note that, by Remark C.1(1), if $\sigma\rho$ is a trace and $\sigma\rho \vdash M$ then $\mathrm{v}(M) \subseteq \widehat{\mathcal{V}}$) and Proposition C.3. □

References

[1] M. Abadi, C. Fournet, Mobile values, new names, and secure communication, in: Conf. Rec. of POPL'01, 2001.

[3] R.M. Amadio, S. Lugiez, On the reachability problem in cryptographic protocols, in: Proc. of Concur'00, Lecture Notes in Computer Science, Vol. 1877, Springer, Berlin, 2000.

[4] R.M. Amadio, S. Lugiez, V. Vanackère, On the symbolic reduction of processes with cryptographic functions, Theoret. Comput. Sci. 290 (1) (2003) 695–740.

[5] R. Amadio, S. Prasad, The game of the name in cryptographic tables, in: Proc. of Asian'00, Lecture Notes in Computer Science, Vol. 1742, Springer, Berlin, 2000. RR 3733 INRIA Sophia Antipolis.

[6] B. Blanchet, An efficient cryptographic protocol verifier based on prolog rules, in: Proc. of 14th Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 2001.

[7] B. Blanchet, A. Podelski, Verification of cryptographic protocols: tagging enforces termination, in: Proc. of FoSSaCS'03, Lecture Notes in Computer Science, vol. 2620, Springer, Berlin, 2003.

[8] M. Boreale, Symbolic trace analysis of cryptographic protocols, in: Proc. of ICALP'01, Lecture Notes in Computer Science, Vol. 2076, Springer, Berlin, 2001.

[9] M. Boreale, M. Buscemi, Experimenting with STA, a tool for automatic analysis of security protocols, in: Proc. of SAC'02, ACM Press, New York, 2002.

[10] M. Boreale, M. Buscemi, A framework for the analysis of security protocol, in: Proc. of CONCUR '02, Lecture Notes in Computer Science, Vol. 2421, Springer, Berlin, 2002.

[12] Y. Chevalier, R. Kuesters, M. Rusinowitch, M. Turuani, An NP decision procedure for protocol insecurity with Xor, in: Proc. of LICS'03, IEEE Computer Society Press, Silverspring, MD, 2003.

[13] Y. Chevalier, R. Kuesters, M. Rusinowitch, M. Turuani, Deciding the security of protocols with Diffie–Hellman exponentiation and product in exponents, in: Proc. of FSTTCS'03, Lecture Notes in Computer Science, Vol. 2914, Springer, Berlin, 2003.

[15] H. Comon, V. Cortier, J. Mitchell, Tree automata with one memory, set constraints and ping-pong protocols, in: Proc. of ICALP'01, Lecture Notes in Computer Science, Vol. 2076, Springer, Berlin, 2001.

[16] H. Comon-Lundh, V. Cortier, New decidability results for fragments of first-order logic and application to cryptographic protocols, in: Proc. of RTA'03, Lecture Notes in Computer Science, Vol. 2706, Springer, Berlin, 2003.

[17] H. Comon-Lundh, V. Shmatikov, Intruder deductions, constraint solving and insecurity decision in presence of exclusive or, in: Proc. LICS '03, IEEE Computer Society Press, Silver Spring, MD, 2003.

[18] D. Dolev, A. Yao, On the security of public-key protocols, IEEE Trans. Inform. Theory 2 (29) (1983) 198–208.

[19] N. Durgin, P. Lincoln, J. Mitchell, A. Scedrov, Undecidability of bounded security protocols, in: Proc. of Workshop on Formal Methods and Security Protocols, 1999.

[20] S. Even, O. Goldreich, On the security of multi-party ping-pong protocols, in: Proc. of FOCS, 1983.

[21] M.P. Fiore, M. Abadi, Computing symbolic models for verifying cryptographic protocols, in: Proc. of 14th Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 2001.

[22] J. Heather, G. Lowe, S. Schneider, How to prevent type flaw attacks on security protocols, in: Proc. of 13th Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 2000.

[23] N. Heintze, J.D. Tygar, A model for secure protocols and their compositions, IEEE Trans. Software Eng. 22 (1) (1996) 16–30.

[24] A. Huima, Efficient infinite-state analysis of security protocols, in: Proc. of Workshop on Formal Methods and Security Protocols, Trento, 1999.

[25] J.W. Lloyd, Foundations of Logic Programming, Springer, Berlin, 1987.

[26] G. Lowe, Breaking and fixing the Needham–Schroeder public-key protocol using FDR, in: Proc. of TACAS'96, Lecture Notes in Computer Science, Vol. 1055, Springer, Berlin, 1996.
[27] G. Lowe, A hierarchy of authentication specifications, in: Proc. of 10th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 1997.
[29] A. Menezes, P.C. van Oorschot, S.A. Vanstone, Handbook of Applied Cryptography, CRC Press, Boca Raton, FL, 1996.
[30] J. Millen, V. Shmatikov, Constraint solving for bounded-process cryptographic protocol analysis, in: Proc. of 8th ACM Conf. on Computer and Communication Security, ACM Press, New York, 2001.
[31] O. Pereira, J.J. Quisquater, A security analysis of the cliques protocols suites, in: Proc. of the 14th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 2001.
[32] J.C. Mitchell, M. Mitchell, U. Stern, Automated analysis of cryptographic protocols using murφ, in: Proc. of Symp. Security and Privacy, IEEE Computer Society Press, Silverspring, MD, 1997.
[33] R. Ramanujam, S.P. Suresh, Tagging makes secrecy decidable for unbounded nonces as well, in: Proc. of FST&TCS'03, Lecture Notes in Computer Science, Vol. 2914, Springer, Berlin, 2003.
[34] R. Ramanujam, S.P. Suresh, A decidable subclass of unbounded security protocols, in WITS'03, 2003.
[35] R. Ramanujam, S.P. Suresh, Undecidability of secrecy for security protocols. Manuscript, 2003. Available at http://www.imsc.res.in/~jam/TR-undec.ps.gz.
[36] A.W. Roscoe, Modelling and verifying key-exchange using CSP and FDR, in: Proc. of 8th Computer Security Foundations Workshop, IEEE Computer Society Press, Silverspring, MD, 1995.
[38] M. Rusinowitch, M. Turuani, Protocol insecurity with finite number of sessions in NP-complete, Theoret. Comput. Sci. 299 (1–3) (2003) 451–475.
[39] S. Schneider, Verifying authentication protocols in CSP, IEEE Trans. Software Eng. 24 (8) (1998) 743–758.
[40] V. Shmatikov. Decidable analysis of cryptographic protocols with products and modular exponentiation, in: Proc. of ESOP '04, Lecture Notes in Computer Science, Vol. 2986, Springer, Berlin, 2004.
[41] STA: a tool for trace analysis of cryptographic protocols. ML object code and examples, 2001. Available at http://www.dsi.unifi.it/~boreale/tool.html. Online version at http://jordie.di.unipi.it:8080/pweb.
[42] F. Thayer, J. Herzog, J. Guttman, Strand spaces: proving security protocols correct, J. Comput. Security 7 (1) (1999) 191–230.