Towards Unconditional Soundness: Computationally Complete Symbolic Attacker

Gergei Bana[1,⋆] and Hubert Comon-Lundh[2,⋆⋆]

[1] NTT Communication Science Laboratories, Atsugi, Kanagawa, Japan
gergei.bana@lab.ntt.co.jp
[2] CNRS, INRIA Project SecSi and LSV, ENS Cachan, France
comon@lsv.ens-cachan.fr

Abstract. We consider the question of the adequacy of symbolic models versus computational models for the verification of security protocols. We neither try to include properties in the symbolic model that reflect the properties of the computational primitives nor add computational requirements that enforce the soundness of the symbolic model. We propose in this paper a different approach: everything is possible in the symbolic model, unless it contradicts a computational assumption. In this way, we obtain unconditional soundness almost by construction. And we do not need to assume the absence of dynamic corruption or the absence of key-cycles, which are examples of hypotheses that are always used in related works. We set the basic framework, for arbitrary cryptographic primitives and arbitrary protocols, however for trace security properties only.

1 Introduction

The automatic analysis of security protocols has been quite successful since 1990, yielding several tools [10,17,23]. However, when the outcome of one of these provers is "the protocol is secure", it must be understood as "secure in our model". Nothing guarantees that the necessary abstractions are relevant to actual implementations. For instance, consider the Needham-Schroder-Lowe protocol [20]. It has been proved secure by all the above-mentioned provers. However, there are several attacks, for instance when the encryption scheme does not guarantee the ciphertext integrity [26] or when the pairing is associative [21] or when some random number could be confused with some pairings [8].

For this reason, it is important to investigate what exactly the assumptions are, on the cryptographic primitives' implementations, that guarantee the faithfulness of the abstraction. (It is called *soundness* in the literature).

There are a lot of works providing some soundness results, typically the works initiated by Backes et al [5,3,6] and Abadi et al [1,15,13]. They essentially prove that a given symbolic model is fully abstract with respect to a computational one, assuming some properties of the security primitives. This guarantees that the security proofs that have been completed in the abstract model are also valid in a computational model.

⋆ Partially supported by FCT project ComFormCrypt PTDC/EIA-CCO/113033/2009.
⋆⋆ Partially supported by the ANR project ProSe.

P. Degano and J.D. Guttman (Eds.): POST 2012, LNCS 7215, pp. 189–208, 2012.

However, these works require a very large set of assumptions, that are not always emphasized. For instance in [7] the complete list of assumptions for public-keys is listed; it is a long list of strong hypotheses, that are not fullfilled by most actual protocols. [13] make even less realistic assumptions, in order to get a stronger soundness result (which includes more security properties). All these results typically assume that no key cycles can ever be created, that bitstrings can be parsed in deterministic polynomial time into terms, that there is no dynamic corruption, that keys are certified, etc. These assumptions, as well as reasons why they are not realistic enough is discussed in [14]. Furthermore, each primitive requires a new soundness proof and each combination of primitives also requires a new soundness proof, unless much stronger properties are assumed [12]. Currently, it seems more realistic to use CryptoVerif [11], completing the proofs directly in the computational model, than using a soundness result [2]. Is it really impossible to avoid manipulating computation times, probabilities, bitstring lengths... ?

In this paper, we advocate a new way of performing proofs in a symbolic, abstract, model, while keeping strong, computational guarantees establishing a general soundness result, but without establishing many specific soundness results for specific properties of primitives. Such properties can later be proven and added.

The idea is to design a symbolic setting, in which any adversarial action is possible, unless it contradicts some axiom expressing a property that must be satisfied under standard computational assumptions. In other words, computational properties, such as IND-CCA, can be (symbolically) axiomatized and added to the system in order to limit the possible adversarial moves. We do not require the axiomatization to be complete. The idea is to only list properties that we know for certain about the implementation, and allow any symbolic move consistent with those properties. In this way, either we find an attack, in which case there is at least one possible set of primitives satisfying the assumed properties and for which the security goal is violated, or the axioms were sufficient to ensure the security of the protocol, in which case any implementation fulfilling these axioms will ensure the security.

This approach has several advantages:

1. Though the proofs are performed in a symbolic setting, they are computationally valid.
2. Thanks to our result (Theorem 2), adding a new cryptographic primitive only requires to design an axiomatization of this primitive and prove it sound due to the cryptographic assumptions: the additional soundness proof is short and modular; it focuses on designated properties instead of considering whole execution models.
3. We may be able prove the security of protocols with weaker assumptions on the primitives. For instance, if we prove the security using only axioms that are sound for IND-CPA encryption, then IND-CPA will be a sufficient hypothesis for security.
4. In each security proof, all assumptions are clearly and formally stated as axioms.
5. In case an attack is found, it may be sufficient to add an axiom (expressing stronger hypotheses on the computational implementation of the primtives) ruling out the attack, then try proving again.
6. We may consider any cryptographic primitive, including XOR for instance (for which there are strong limitations of the computational soundness approach [4,25]). Dynamic corruption, key cycles, etc. are not a priori discarded.

Related works. The most closely related works are probably those that consider a proof system that is sound w.r.t. the computational semantics, such as [16,8]. Though these works are related, as far as the computational semantics of the logic is concerned, the overal strategy is completely different. We do not try to design a proof system working directly in the computational model: we only use first-order logic and standard inference rules in the symbolic model. Our approach is more inspired by circumscription [19], however circumscribing what is *not* possible. In other words, we do not design inference rules, we modify the transition system instead. This is similar in spirit to [24], in which any property of the hash function, that is not explicitly forbidden by some axiom, is considered as valid.

Contents of the paper. In this paper, we only state the framework of the method, prove a general soundness theorem in the case of trace properties, and prove soundness of an example axiom expressing secrecy of an IND-CCA encryption.

More precisely, protocols are identified to a formal transition system in the same spirit as CoSP [7]: we do not commit to a very particular way of specifying such a transition system. The possible transitions are, roughly, defined by a formula, that guards the transition by constraining the input message, a state move and a message that is sent out when the guard is satisfied. Such transitions can be interpreted in different models: symbolic models, in which the messages are terms and the guards are interpreted in a Herbrand model, or computational models, in which messages are bitstrings. In the symbolic models, we constrain the input messages to be *deducible* from the previous outputs and the public information. Such a deducibility condition is formalised using a deducibility predicate, whose interpretation is not fixed. This is a main difference with classical protocol verification: the attacker capabilities are not fixed, but rather they parametrize the model. Actually, we consider any attacker capability, that does not contradict the (computationally sound) axioms. On the computational side, the attacker is any probabilistic polynomial time Turing machine: the deduction capabilities are given by any such machine. These models are explained in the sections 2.2, 2.3, 2.5.

Next, we need to speficy the axioms and the (trace) security properties. We consider any first-order formula, that is built on the predicate symbols, that are used in the guards, as well as the deducibility predicate symbol. We need such general formulas, since we need to constrain the symbolic models of the deducibility relations, i.e., the symbolic attacker capabilities, according to the computational assumptions on the primitives. Typically, we may consider an axiom of the form: "if a plaintext can be deduced (resp. computed) from a ciphertext and a set of messages ϕ, then the decryption key has been sent out or else the plaintext can be deduced (resp. computed) from ϕ", that reflects some property of the encryption scheme. The meanings of these axioms/security properties become clear when we define a computational interpretation of such formulas, which we provide in the section 2.6.

The Section 3 is devoted to the main result, which states a general trace-mapping soundness property: independently of the primitives and their specific characteristics, if there is a computational attack, then there is a symbolic attack. Once more, the symbolic attacker has any capability, that is consistent with the axioms. So, this result, though subtle and not at all trivial to prove, is not surprising. The whole system was actually carefully designed with this aim in mind.

We also show in the Section 4 some axiom examples, that are proven sound under some standard cryptographic properties. We do not aim however at covering a large set of axioms. Further axioms will be added to a library each time they are required for the proof of a case study.

This paper aims at opening a new research direction: it seems very appealing and promising. We need however to investigate several case studies. As a "proof of concept", we have designed a complete set of axioms and proved the NSL protocol in our framework (available from the first-author's web page or upon request). This sufficient set of axioms shows also that some hypotheses of earlier works are not necessary (at least for weak secrecy and authentication).

2 Symbolic and Computational Models

2.1 Terms and Frames

Terms are built out of a set of function symbols $\mathcal{F}$ that contains an unbounded set of names $\mathcal{N}$ and an unbounded set of handles $\mathcal{H}$. Let $\mathcal{X}$ be an unbounded set of variables. Names and handles are zero arity function symbols. We will use names to denote items honestly generated by agents, while handles will denote inputs of the adversary. A ground term is a term without variables. *Frames* are sequences of terms together with name binders: a frame ϕ can be written $(\nu\overline{n}).p_1 \mapsto t_1, \ldots, p_n \mapsto t_n$ where $p_1, \ldots p_n$ are place holders that do not occur in $t_1, \ldots, t_n$ and $\overline{n}$ is a sequence of names. $\mathsf{fn}(\phi)$, the *free names* of ϕ are names occurring in some t_i and not in $\overline{n}$. The *variables* of ϕ are the variables of $t_1, \ldots, t_n$.

Example 1. We typically use a randomized public-key encryption symbol: $\{m\}^r_{eK_Q}$ is intended to represent the encryption of the plaintext m with the public-key of the principal Q, with a random seed r. More generally, we consider the example when there is a set of constructors $\mathcal{F}_c = \{\{\_\}^{\_}_{\_}, \langle \_, \_ \rangle, e_\_, d_\_, K_\_\}$, and a set of destructors $\mathcal{F}_d = \{dec(\_, \_), \pi_1(\_), \pi_2(\_)\}$, and $\mathcal{F} = \mathcal{F}_c \cup \mathcal{F}_d \cup \mathcal{N} \cup \mathcal{H}$.

2.2 Formulas

Let $\mathcal{P}$ be a set of predicate symbols over tems. $\mathcal{P}$ is assumed to contain the equality $=$ (which is interpreted as a congruence), used as $t_1 = t_2$, and a predicate $\vdash$, which takes as arguments an n-tuple of terms on its left and a term on its right (and which is intended to model the computation capabilities), that is, written as $t_1, ..., t_n \vdash t$. (More precisely, it is an infinite sequence of predicates, with arguments $n + 1$.)

We are not interested in any specific symbolic interpretation of these predicate symbols. We wish to consider *any* possible symbolic interpretation, that satisfies some requirements; the aim is to allow anything that is not forbidden by explicit assumptions.

Example 2. $\forall x, \forall y.(\{x\}^s_{eK_Q} = \{y\}^{s'}_{eK_Q} \rightarrow x = y)$ is such a formula, the validity of which follows from the uniqueness of decryption.

Let $\mathcal{M}$ denote then any first-order structure that interprets the function and predicate symbols of the logic. We only assume that $=$ is interpreted in $\mathcal{M}$ as the equality in the underlying domain $D_{\mathcal{M}}$. The relation in $\mathcal{M}$ (that is, a relation for elements in $D_{\mathcal{M}}$), interpreting the deducibility predicate $\vdash$ is denoted as $\vdash_{\mathcal{M}}$.

Given an assigment σ of elements in $D_{\mathcal{M}}$ to the free variables of term t, we write $[\![t]\!]^{\sigma}_{\mathcal{M}}$ for the interpretation of t in $\mathcal{M}$ ($[\![\_]\!]^{\sigma}_{\mathcal{M}}$ is the unique extension of σ into a homomorphism of $\mathcal{F}$-algebras).

For any first order structure $\mathcal{M}$ over the functions $\mathcal{F}$ and predicates $\mathcal{P}$, given a first order formula θ and an assignment σ of elements in the domain of $\mathcal{M}$ to the free variables of θ, the satisfaction relation $\mathcal{M}, \sigma \models \theta$ is defined as usual in first-order logic.

Example 3. Consider the public-key encryption setting of example 1. We may use unary predicate symbols to restrict sets of data. Assume for instance that W is supposed to represent the set of agent names, and M is supposed to represent well formed terms (that are equal to a term built with symbols in $\mathcal{F}_c$).

$$W(\pi_1(\mathit{dec}(h, db))) \wedge M(\pi_2(\mathit{dec}(h, db)))$$

is a formula, that expresses that the handle h can be decrypted and projected into two components, one of which is an agent name.

2.3 Protocols

We do not stick to any particular syntax for the definition of protocols. We only assume that it defines a transition system as follows. Q is a set of *control states*, together with a finite set of free variables.

Definition 1. *A* protocol *is a recursive (actually PTIME) set of tuples*

$$(q(\overline{n}), q'(\overline{n} \cdot \overline{n'}), \langle x_1, \ldots, x_k \rangle, x, \psi, s)$$

where $q, q' \in Q$, $x_1, \ldots, x_k, x$ are variables $\overline{n}, \overline{n'}$ are finite sequences of names, ψ is a first order formula over the set of predicate symbols $\mathcal{P}$ and function symbols $\mathcal{F}$ and the names $\overline{n} \cup \overline{n'}$, whose free variables are in $\{x_1, \ldots, x_n, x\}$ and s is a term whose free variables are in $\{x_1, \ldots, x_n, x\}$.

For example, ψ can be a formula such as $\mathit{dec}(x, k) = n$, that checks that the current input is a ciphertext whose plaintext is a previously generated nonce n: ψ guards the transition. s is the output message, when the transition succeeds. The intended meaning of these rules is that a transition from the sate q to the state q' is possible, given the previous inputs $x_1, \ldots, x_n$ and the new input x, if the formula ψ is satisfied. In such a case, the names $\overline{n'}$ are generated and the message s is sent.

Such a formalism is quite general; we only assume here (for simplicity) a single, public, communication channel. Typically, applied π-calculus processes can be translated into such transition rules, that are similar to the CoSP framework of [7].

Example 4. Consider a single session of the NSL protocol. The states consist of pairs of the local states of each of the processes for A and B. Instead of listing the transitions as tuples, we write $\psi : q(\overline{n}) \xrightarrow{s} q'(\overline{n})$ and they are diplayed in the figure 1. In this version of the protocol, the responder is willing to communicate with anybody, hence only checks $W(\pi_1(dec(y, dK_B)))$; the intended meaning of W is a set of agent names. If

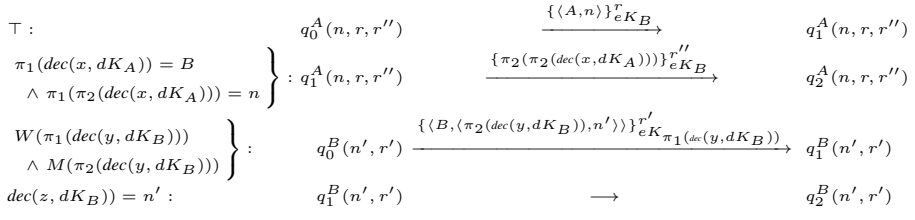

Fig. 1. The 3 transitions of 1 session of NSL

we wish to describe an unbounded number of sessions, we need to record in the control state the states of every (opened) A-session and (opened) B-session. This yields an infinite, yet recursive, set of transition rules.

Definition 2. *A* symbolic state *of the network consists of:*

- *a control state $q \in Q$ together with a sequence of names (that have been generated so far) $n_1, \ldots, n_k$*
- *a sequence constants called* handles *$h_1, \ldots, h_n$ (recording the attacker's inputs)*
- *a ground frame ϕ (the agents outputs)*
- *a set of formulas Θ (the conditions that have to be satisfied in order to reach the state).*

A *symbolic transition sequence* of a protocol Π is a sequence

$$(q_0(\overline{n_0}), \emptyset, \phi_0, \emptyset) \to \ldots \to (q_m(\overline{n_m}), \langle h_1, \ldots, h_m\rangle, \phi_m, \Theta_m)$$

if, for every $m - 1 \geq i \geq 0$, there is a transition rule

$$(q_i(\overline{\alpha_i}), q_{i+1}(\overline{\alpha_{i+1}}), \langle x_1, \ldots, x_i\rangle, x, \psi, s)$$

such that $\overline{n} = \overline{\alpha_{i+1}} \setminus \overline{\alpha_i}$, $\phi_{i+1} = (\nu \overline{n}).(\phi_i \cdot p \mapsto s\rho_i\sigma_{i+1})$, $\overline{n_{i+1}} = \overline{n_i} \uplus \overline{n}$, $\Theta_{i+1} = \Theta_i \cup \{\phi_i \vdash h_{i+1}, \psi\rho_i\sigma_{i+1}\}$ where $\sigma_i = \{x_1 \mapsto h_1, \ldots, x_i \mapsto h_i\}$ and ρ_i is a renaming of the sequence $\overline{\alpha_i}$ into the sequence $\overline{n_i}$. We assume a renaming that ensures the freshness of the names $\overline{n}$: $\overline{n} \cap \overline{n_i} = \emptyset$.

Definition 3. *Given an interpretation $\mathcal{M}$, a transition sequence of Π*

$$(q_0(\overline{n_0}), \emptyset, \phi_0, \emptyset) \to \ldots \to (q_m(\overline{n_m}), \langle h_1, \ldots, h_m\rangle, \phi_m, \Theta_m)$$

is valid w.r.t. *$\mathcal{M}$ if, for every $m - 1 \geq i \geq 0$,*

$$\mathcal{M} \models \Theta_{i+1}$$

Example 5. We show the beginning of a possible branch in the symbolic execution of NSL.

$$(q_0, \emptyset, \phi_0, \emptyset) \quad (q_1, H_1, \phi_1, \Theta_1) \; (q_2, H_2, \phi_2, \Theta_2) \; (q_3, H_3, \phi_3, \Theta_3) \; (q_4, H_4, \phi_4, \Theta_4)$$

Where $\overline{n} = n, r, r'', n', r', q_0 = (q_0^A, q_0^B)(\overline{n})$, and $q_1 = (q_1^A, q_0^B)(\overline{n})$, $q_2 = (q_1^A, q_1^B)(\overline{n})$, and $q_3 = (q_2^A, q_1^B)(\overline{n})$ and $q_4 = (q_2^A, q_2^B)(\overline{n})$. In other words, we interleave the actions of A and B, as in an expected execution and assume that the two processes were first activated (if not, we could introduce two transitions activating the processes).

- $\phi_0 = \nu_{K_A K_B AB}(p_0 \mapsto (A, B, eK_A, eK_B))$,
  $\Theta_0 = \emptyset$
- $H_1 = \langle h_1 \rangle$,
  ϕ_1 extends ϕ_0 with $p_1 \mapsto \{\langle A, n\rangle\}^r_{eK_B}$,
  $\Theta_1 = \{\phi_0 \vdash h_1\}$
- $H_2 = \langle h_1, h_2 \rangle$,
  ϕ_2 extends ϕ_1 with $p_2 \mapsto \{\langle B, \langle \pi_2\,(dec(h_2, dK_B)), n'\rangle\rangle\}^{r'}_{eK_{\pi_1(dec(h_2, dK_B))}}$,
  $\Theta_2 = \Theta_1 \cup \{\phi_1 \vdash h_2, M(\pi_2\,(dec(h_2, dK_B))), W(\pi_1\,(dec(h_2, dK_B)))\}$
- $H_3 = \langle h_1, h_2, h_3 \rangle$,
  ϕ_3 extends ϕ_2 with $p_3 \mapsto \{\pi_2\,(\pi_2\,(dec(h_3, dK_A)))\}^{r''}_{eK_B}$,
  $\Theta_3 = \Theta_2 \cup \{\phi_2 \vdash h_3, \pi_1\,(\pi_2\,(dec(h_3, dK_A))) = n, \pi_1\,(dec(h_3, dK_A)) = B\}$,
- $H_4 = \langle h_1, h_2, h_3, h_4 \rangle$, $\phi_4 = \phi_3$,
  $\Theta_4 = \Theta_3 \cup \{\phi_3 \vdash h_4, dec(h_4, dK_B)) = n'\}$,

Let $\mathcal{M}$ be a model in which $\pi_1\,(dec(h_2, dK_B)) = A$ and

$$h_2 =_{\mathcal{M}} \{\langle A, n\rangle\}^r_{eK_B}, \qquad h_3 =_{\mathcal{M}} \{\langle B, \langle n, n'\rangle\rangle\}^{r'}_{eK_A}, \qquad h_4 =_{\mathcal{M}} \{n'\}^{r''}_{eK_B},$$

and $\vdash_{\mathcal{M}}$ is simply the classical Dolev-Yao deduction relation. Then the execution sequence is valid w.r.t. $\mathcal{M}$, and this corresponds to the correct execution of the NSL protocol between A and B.

There are however other models in which this transition sequence is valid. For instance let $\mathcal{M}'$ be such that $h_2 =_{\mathcal{M}'} n$ and $\phi_1 \vdash_{\mathcal{M}'} n$ and $n =_{\mathcal{M}'} \{\langle A, n\rangle\}^r_{eK_B}$, (and h_3, h_4 as above). We get again a valid transition sequence w.r.t. $\mathcal{M}'$. Though, in what follows, we will discard such sequences, thanks to some axioms.

Example 6. Consider again the transitions of the example 5. Now consider a model $\mathcal{M}$ in which $n_0, \{B, n, n'\}^r_{eK_A} \vdash_{\mathcal{M}} \{B, n_0, n'\}^r_{eK_A}$ for an honestly generated nonce n_0 that can be chosen by the attacker: the transition sequence of the previous example is also valid w.r.t. this model. This will yield an attack, using a malleability property of the encryption scheme, as in [26]. Discarding such attacks requires some properties of the encryption scheme (for instance IND-CCA). It can be ruled out by a non-malleability axiom (the discussion of which is out of the scope of this paper, but included in the NSL proof referred to in the introduction).

From these examples, we see that unexpected attacks can be found when some assumption is not explicitly stated as an axiom to limit adversarial capabilities.

2.4 Axioms and Security Properties

For simplicity, we only consider reachability security properties. The extension to any trace property should not be very difficult: it suffices to record some values along the trace. Security properties (and, later, axioms) are first-order formulas that may contain state-dependent predicates and/or predicates that get fixed interpretation. As in the previous sections, $\mathcal{M}$ is an arbitrary first-order structure and σ is an assignment of the free variables to elements of $D_{\mathcal{M}}$.

First, we add atomic formulas $\hat{\phi}, s_1, \ldots, s_n \vdash t$, where $\hat{\phi}$ is just part of the syntax of this predicate (not an input of the predicate), which aims at ranging over frames (when interpretating the predicate) and is evaluated in every state. For $t_1, ..., t_m$ closed terms, $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m \rangle, \overline{n} \models \hat{\phi}, s_1, \ldots, s_n \vdash t$ iff $\mathcal{M}, \sigma \models s_1, \ldots, s_n, t_1, \ldots, t_m \vdash t$.

In addition, we consider the following atomic formulas, whose evaluation only depends on the state, independently of the first-order structure $\mathcal{M}$.

- $\mathsf{RandGen}(s)$ (s is a ground term) expresses that s has been randomly generated: $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m \rangle, (n_1, \ldots, n_k) \models \mathsf{RandGen}(s)$ iff $s \in \{n_1, \ldots, n_k\}$
- $t \sqsubseteq \hat{\phi}$ (t is a ground term) expresses that t is a subterm of the messages sent so far: $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m \rangle, \overline{n} \models t \sqsubseteq \hat{\phi}$ iff t is a subterm of some t_i.
- We also may use the derived predicate (as an abbreviation):

$$\mathsf{fresh}(x, \hat{\phi}) = \mathsf{RandGen}(x) \wedge x \not\sqsubseteq \hat{\phi}$$

$\sqsubseteq$ and $\mathsf{RandGen}()$ are *interpreted predicates* since their interpretation does not depend on $\mathcal{M}$. Bound variables that appear within an interpreted predicate are called *constrained variables*. As in other works on constrained logics (see for instance [18]), such variables are used to schematize several first-order formulas and are replaced with ground terms built on $\mathcal{F}$. Therefore, the interpretation of axioms and security properties that may involve interpreted predicates, is modified, only in case of a quantification on a constrained variable x, in which case x is replaced by any (or some, for existential quantification) ground term:

If x is a constrained variable (that is, θ has an interpreted predicate and x appears in it), then,

$$\mathcal{M}, \sigma, \langle t_1, \ldots, t_n \rangle, (n_1, \ldots, n_k) \models \forall x.\theta$$

iff, for every ground term t,

$$\mathcal{M}, \sigma, \langle t_1, \ldots, t_n \rangle, (n_1, \ldots, n_k) \models \theta\{x \mapsto t\}$$

We have a similar definition for existential quantifications of such variables. All other cases follow the classical definition of the first-order satisfaction relation.[1] This yields a satisfaction relation $\mathcal{M}, \sigma, \langle t_1, \ldots, t_m \rangle, \overline{n} \models \theta$, and thus of $\mathcal{M}, \sigma, \phi, \overline{n} \models \theta$ with ϕ

[1] It would in fact be possible to avoid the notion of constrained variables if we defined $D_{\mathcal{M}}$ to be a free $\mathcal{F}$-algebra, and $=$ a congruence relation on it (as opposed to the equality of $D_{\mathcal{M}}$), and later parts of the paper could be adjusted accordingly. However, since constrained variables are more convenient for automatic verification, the authors decided to present the theory utilizing them.

having the terms $\langle t_1, \ldots, t_m \rangle$. When θ has no free variable, we may omit σ. Similarly, if θ does not contain atomic formulas that depend on ϕ (resp. $\overline{n}$), we may omit these components: we get back to the satisfaction relation of section 2.2.

We define now the satisfaction relation in a state:

$$\mathcal{M}, (q, \langle h_1, \ldots, h_m \rangle, \overline{n}, \phi_m, \Theta) \models \theta \quad \text{iff} \quad \mathcal{M}, \phi_m, \overline{n} \models \theta.$$

Definition 4. *A symbolic interpretation and a protocol* satisfy the security property θ, *written as*

$$\mathcal{M}, \Pi \models \theta,$$

if for any sequence of transitions that is executable in $\mathcal{M}$ and that yields the state $(q_m, \langle h_1, \ldots, h_m \rangle, \overline{n_m}, \phi_m, \Theta_m)$,

$$\mathcal{M}, (q_m, \langle h_1, \ldots, h_m \rangle, \overline{n_m}, \phi_m, \Theta_m) \models \theta.$$

Example 7. Concerning security properties, consider the NSL protocol. We may state the confidentiality of n:

$$\neg \hat{\phi} \vdash n$$

Consider now an authenticity property. We modify slightly the states of the transition system, including a commitment on the nonce on which the parties are supposed to agree. We let c_i be a special function symbol, that takes as arguments A, B, n_1, n_2: who commits, for who and the corresponding nonces. $c_i(A, B, n_1, \pi_2(\pi_2(\mathit{dec}(x, dK_B))))$ is sent at the end by the initiator. For the responder, there is a similar commitment: at the end of the protocol, B emits $c_r(\pi_1(\mathit{dec}(x, dK_B)), B, \pi_2(\mathit{dec}(y, dK_B), n_2))$. We state as axioms that c_i, c_r cannot help the attacker and cannot be forged. For instance: $\forall x, y, z, w. \hat{\phi}, c_i(x, y, z, w) \not\vdash z, w$ and $\forall x, y, z, w. \hat{\phi} \vdash c_i(x, y, z, w) \rightarrow c_i(x, y, z, w) \sqsubseteq \hat{\phi}$. The agreement property (on n) may then be stated (for instance) as:

$$\forall x, y, z, w. c_r(x, y, z, w)) \sqsubseteq \hat{\phi} \rightarrow \exists x' z' w' (c_i(x', y, z', w') \sqsubseteq \hat{\phi} \wedge x = x' \wedge z = z' \wedge w = w')$$

That is: x's view of z, w is the same as y's view of z, w.

With such a definition, for any security property and any protocol there will (almost) always be an interpretation for which the property is violated. Hence we restrict the class of symbolic interpretations, ruling out the interpretation whose all computational counterparts would violate some security assumption on the primitives. More precisely, we consider a set of *axioms* $\mathcal{A}$, which is a set of first-order formulas in the same format as the security properties. We restrict our attention to symbolic interpretations that satisfy $\mathcal{A}$.

Example 8.

- For instance we could include in $\mathcal{A}$ a formula

$$\mathsf{fresh}(k, \hat{\phi}) \rightarrow \neg(\hat{\phi} \vdash k)$$

that states that an attacker cannot guess (except with negligible probability) a randomly generated name. Adding such an axiom in $\mathcal{A}$ rules out symbolic interpretations in which this deduction is possible.

– If the computational implementation is such (e.g. they are tagged), we may include,

$$\forall x, y, z, A, r. \langle x, y \rangle \neq \{z\}^r_{K_A}$$

stating that pairs and ciphertexts cannot be confused.

We will see more examples in Section 4.

We may assume w.l.o.g that the axioms and security properties are just (universally quantified) clauses.

2.5 Computational Interpretation

The computational interpretations are just a special case of interpretation of our formulas, when they do not depend on the state of the transition system. We define them again here, since we wish to introduce some additional notions. Also, the computational executions of the protocols rely on a concrete adversary, given by a Turing machine, while in general, the interpretation of functions and predicates need not to be computable.

We consider a familly computational algebras, parametrized by a security parameter η, in which each function symbol is interpreted as a polynomially computable function on bitstrings (that may return an error message). Given then a sample τ of names (for every name n, its interpretation is a bitstring $\tau(n)$), every ground term t is interpreted as a bitstring $[\![t]\!]_\tau$ in such a way that $[\![\_]\!]_\tau$ is a homomorphism of $\mathcal{F}$-algebras. More generally, if σ is an assignment of the variables of t to bitstrings $[\![t]\!]^\sigma_\tau$ is the (unique) extension of τ (on names) and σ (on variables) as a homomorphism of $\mathcal{F}$-algebras.

Similarly, all predicate symbols are interpreted as polynomially computable functions on bitstrings. The equality predicate is interpreted as a strict equality on bitsrings: $\tau \models^c t_1 = t_2$ if $[\![t_1]\!]_\tau$ is not an error, $[\![t_2]\!]_\tau$ is not an error and $[\![t_1]\!]_\tau = [\![t_2]\!]_\tau$.

This interpretation is extended to arbitrary closed formulas whose atomic formulas do not depend on the state. This yields the satisfaction relation $\tau \models^c \theta$. We will define later the computational interpretation of arbitrary formulas in a given state.

We now define computational executions.

Definition 5. *Given a set of transition rules, a* computational state *consists of*

– *A symbolic state s (that is itself a tuple $q(\overline{n}, \overline{h}, \phi, \Theta)$)*
– *a sequence of bitstrings $\langle b_1, \ldots, b_m \rangle$ (the attacker's outputs)*
– *A sequence $\langle b'_1, \ldots, b'_n \rangle$ of bitstrings (the agents outputs)*
– *The configuration γ of the attacker.*

Definition 6. *Given a PPT interactive Turing machine $\mathcal{M}$ and a sample τ, a sequence of transitions*

$$(s_0, \emptyset, \boldsymbol{b'_0}, \gamma_0) \to \ldots \to (s_m, \langle b_1, \ldots, b_m \rangle, \langle b'_1, \ldots, b'_m \rangle, \gamma_m)$$

is (computationally) valid w.r.t. $\mathcal{M}$ and τ *if*

– *$s_0 \to \cdots \to s_m$ is a transition sequence of the protocol*
– *for every $i = 0, ...m-1$, $s_i = (q_i(\overline{n_i}), \overline{h_i}, \phi_i, \Theta_i)$, $\phi_{i+1} = (\nu \overline{n}).\phi_i \cdot u_i$, $[\![u_i]\!]_\tau = b'_{i+1}$*

- *for every* $i = 0, ..., m-1$*, there is a configuration* γ'_i *of the machine* $\mathcal{M}$ *such that* $\gamma_i \vdash^*_M \gamma'_i \vdash^*_M \gamma_{i+1}$ *and* γ'_i *is in a sending state, the sending tape containing* b_{i+1}*,* γ_{i+1} *is in a receiving state, the receiving tape containing* b'_{i+1}
- *for every* $i = 0, ..., m-1$*,* $\tau, \{x_1 \mapsto b_1, \ldots, x \mapsto b_{i+1}\} \models^c \Theta_{i+1}$.

Intuitively, $\boldsymbol{b'}_0$ is the attacker's initial knowledge and we simply replaced symbolic deductions/symbolic models of the section 2.3 with computations/computational models.

2.6 Computational Validity of Security Properties and Axioms

We already considered the computational satisfaction of formulas, except for formulas that depend on the states. Given a PT Turing machine $\mathcal{A}$, we define then

$$\mathcal{A}, \tau \models^c t_1, \ldots, t_n \vdash t \quad \text{iff} \quad \mathcal{A}([\![t_1]\!]_\tau, \ldots, [\![t_n]\!]_\tau) = [\![t]\!]_\tau$$

The difficulty now is that we do not want to define $\mathcal{A}, \tau \models^c \hat{\phi} \vdash t_1 \rightarrow \hat{\phi} \vdash t_2$ as $\mathcal{A}, \tau \models \hat{\phi} \vdash t_2$ or $\mathcal{A}, \tau \not\models \hat{\phi} \vdash t_1$. In order to understand this, consider for instance the formula

$$\theta: \quad \forall_{t,K,R}(\hat{\phi}, \{t\}^R_{eK} \vdash t \rightarrow \{t\}^R_{eK} \sqsubseteq \hat{\phi} \vee dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t)$$

We want (intuitively) IND-CCA encryption schemes to satisfy this formula. However, consider an instance of this axiom in which $\hat{\phi}$ is the pair $\phi = \nu n_1 n_2.\langle n_1, n_2\rangle$, and t is n_1. Now, let $\mathcal{A}$ be a machine which, on input $[\![\langle n_1, n_2\rangle]\!]_\tau, [\![\{n_1\}^r_{eK}]\!]_\tau$ returns n_1 and, on input $[\![\langle n_1, n_2\rangle]\!]_\tau$ only, returns $[\![n_2]\!]_\tau$. For every τ, $\mathcal{A}, \tau \not\models^c \theta$. Hence, whatever security is provided by the encryption scheme, there is an attack on the property.

This paradox comes from the deterministic interpretation of the deducibility relation: while, symbolically, it is a relation, it must be a function in the computational setting since we cannot consider non-deterministic machines. The intended interpretation therefore involves several machines: roughly, for any machine that can compute $[\![t]\!]_\tau$ from $[\![\phi]\!]_\tau, [\![\{t\}^r_{eK}]\!]_\tau$, either there is a machine that can compute $[\![t]\!]_\tau$ from $[\![\phi]\!]_\tau$ or else the actual frame contains either dK or $\{t\}^r_{eK}$. These two machines need of course to be independent of τ. This is the definition that we formalize now for arbitrary security properties.

Let $\mathcal{M}$ be an interactive PPT Turing machine with a special challenge control state q_c. We may regard this machine as an attacker, who moves to the state q_c when (s)he thinks that (s)he is ready to break the security property.

In what follows, S is any (polynomial time) non-negligible set of interpretations of names, and $S_\top$ is the set of all name interpretations. $\mathcal{M}, \Pi \models^c \theta$ iff $\mathcal{M}, \Pi, S_\top \models^c \theta$ and $\Pi \models^c \theta$ if $\mathcal{M}, \Pi \models^c \theta$ for every $\mathcal{M}$ with q_c.

We introduce machines that compute witnesses for the unconstrainted quantified variables.

- $\mathcal{M}, \Pi, S \models^c \exists x.\theta$ iff there is a PT machine $\mathcal{A}_x$ such that $\mathcal{M}, \Pi, S, \mathcal{A}_x \models^c \theta$
- $\mathcal{M}, \Pi, S, \mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n} \models^c \forall x.\theta$ iff for any probabilistic polynomial time machine $\mathcal{A}_x$, $\mathcal{M}, \Pi, S, \mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n}, \mathcal{A}_x \models^c \theta$

If x is a constrained variable, the interpretation of $\forall x.\theta$ is analogous to the symbolic case: $\mathcal{M}, \Pi, S, \mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n} \models^c \forall x.\theta$ if and only if for every ground term t, the satisfaction $\mathcal{M}, \Pi, S, \mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n} \models^c \theta\{x \mapsto t\}$ holds (and similarly for existential quantification). If σ is a sequence of machines, one for each free variable x of θ,

- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \wedge \theta_2$ iff $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1$ and $\mathcal{M}, \Pi, S, \sigma \models^c \theta_2$.
- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \vee \theta_2$ iff there are sets $S_1 \cup S_2 = S$ such that $\mathcal{M}, \Pi, S_1, \sigma \models^c \theta_1$ and $\mathcal{M}, \Pi, S_2, \sigma \models^c \theta_2$.
- $\mathcal{M}, \Pi, S, \sigma \models^c \theta_1 \rightarrow \theta_2$ iff for any $S' \subseteq S$ non-negligible, $\mathcal{M}, \Pi, S', \sigma \models^c \theta_1$ implies $\mathcal{M}, \Pi, S', \sigma \models^c \theta_2$
- $\mathcal{M}, \Pi, S, \sigma \models^c \neg\theta$ iff for any S', if $\mathcal{M}, \Pi, S', \sigma \models^c \theta$, then $S \cap S'$ is negligible
- in the case of an atomic formula $P(t_1, \ldots, t_n)$ where $P \notin \{\vdash, \sqsubseteq, \mathsf{RandGen}()\}$, $\mathcal{M}, \Pi, S, \sigma \models^c P(t_1, \ldots, t_n)$ if there is an overwhelming subset S' of S such that the following holds. For any $\tau \in S'$, consider the unique valid computation (if there is one) of Π with respect to $\mathcal{M}, \tau$ that yields a configuration of $\mathcal{M}$, which is in the control state q_c with a bitstring b on the output tape. Let $q(\overline{n})$ be the control state reached at this point and $\boldsymbol{c}$ be the restriction of τ to $\overline{n}$. Let $b_x = \mathcal{A}_x(b, \boldsymbol{c})$ for every $\mathcal{A}_x \in \sigma$, and α be the sequence of assignments $x \mapsto b_x$. Then $(\llbracket t_1 \rrbracket^\alpha_\tau, \ldots, \llbracket t_n \rrbracket^\alpha_\tau) \in \llbracket P \rrbracket$.
- For the deducibility predicate, $\mathcal{M}, \Pi, S, \sigma \models^c \hat{\phi}, t_1, \ldots, t_n \vdash t$ if for all non-negligible $S' \subseteq S$, there is a non-negligible $S'' \subseteq S'$ such that there is a PT Turing machine $\mathcal{A}_D$ such that, for all $\tau \in S''$, the unique valid computation (if there is one) of Π with respect to $\mathcal{M}, \tau$ that yields a configuration of $\mathcal{M}$, which is in the control state q_c with a bitstring b on the output tape, an actual frame ϕ_m and such that, letting $b_x = \mathcal{A}_x(b, \overline{c})$ where $\overline{c} = \tau(\overline{n})$ for the names $\overline{n}$ in the current state, for every $\mathcal{A}_x \in \sigma$, and α be the sequence of assignments $x \mapsto b_x$, $\mathcal{A}_D(\llbracket \phi_m \rrbracket_\tau, \llbracket t_1 \rrbracket^\alpha_\tau, \ldots, \llbracket t_n \rrbracket^\alpha_\tau, b) = \llbracket t \rrbracket^\alpha_\tau$.
- $\mathcal{M}, \Pi, S, \sigma \models^c t_1, \ldots, t_n \vdash t$ is defined exactly as above, however removing ϕ.
- If P is an interpreted predicate, $\mathcal{M}, \Pi, S, \sigma \models P(t_1, \ldots, t_n)$ iff there is an overwhelming subset $S' \subseteq S$ such that, for any $\tau \in S'$, the unique valid computation of Π with respect to $\mathcal{M}, \tau$ that yields a computational state $(q(\overline{n}, \overline{h}, \phi, \Theta), \overline{b}, \overline{b'}, \gamma)$ in the control state q_c such that $P(t_1, \ldots, t_n)$ is true in $q(\overline{n}, \overline{h}, \phi, \Theta)$. (Remember that the evaluation of such predicates do not depend on the model).

$\mathcal{M}, \Pi \models^{nnp} \theta$ (read "$\mathcal{M}, \Pi$ satisfies θ with non negligible probability") if there is a non-negligible set S and a PPT machine $\mathcal{A}$ such that $\mathcal{A}(n_1, \ldots, n_k, b_1, \ldots, b_k)$ returns 1 iff there is a $\tau \in S$ such that, for all i, $\tau(n_i) = b_i$ and $\mathcal{M}, \Pi, S \models^c \theta$.

Lemma 1. *If $\mathcal{M}, \Pi, S, \sigma \models^c \theta$ and $S' \subseteq S$, then we also have $\mathcal{M}, \Pi, S', \sigma \models^c \theta$.*

Proof. We proceed by induction on θ. Since $\mathcal{M}, \Pi$ are fixed, we sometimes omit these components.

- If θ is a atomic formula $P(t_1, \ldots, t_n)$ and $P \notin \{\vdash, \mathsf{fresh}(), \sqsubseteq\}$, then, by definition, there is an overwhelming subset $S_1 \subseteq S$ such that, for any $\tau \in S_1$, $(\llbracket t_1 \rrbracket^\alpha_\tau, \ldots, \llbracket t_n \rrbracket^\alpha_\tau) \in \llbracket P \rrbracket$. If $S' \subseteq S$, we choose $S'_1 = S' \cap S_1$. It is an overwhelming subset of S' and, for any $\tau \in S'_1$, $(\llbracket t_1 \rrbracket^\alpha_\tau, \ldots, \llbracket t_n \rrbracket^\alpha_\tau) \in \llbracket P \rrbracket$.

- If θ is a formula $\hat{\phi}, t_1, \ldots, t_n \vdash t$, then for any non negligible $S_1 \subseteq S$, there is a non-negligible $S_2 \subseteq S_1$ and there is a machine $\mathcal{A}$, such that, for any $\tau \in S_2$, $\mathcal{A}([\![\phi]\!]_\tau, [\![t_1]\!]^\alpha_\tau, \ldots, [\![t_n]\!]^\alpha_\tau) = [\![t]\!]^\alpha_\tau$. If $S' \subseteq S$ is non negligible, then any non-negligible $S'_1 \subseteq S'$ is also a non-negligible $S'_1 \subseteq S$, hence the result.
- Other atomic formulas with $=$ and $\sqsubseteq$ are rather trivial.
- If $\theta = \neg\theta_1$, $\mathcal{M}, \Pi, S, \sigma \models^c \theta$ iff for any $S_1 \subseteq S$ such that $\mathcal{M}, \Pi, S_1, \sigma \models^c \theta_1$, $S_1 \cap S$ is negligible. In that case, for any $S' \subseteq S$, $S' \cap S_1$ is also negligible, hence the result (we do not use here the induction hypothesis).
- If $\theta = \theta_1 \vee \theta_2$, $S, \sigma \models^c \theta_1 \vee \theta_2$ implies $S = S_1 \cup S_2$ and $S_1, \sigma \models^c \theta_1$, $S_2, \sigma \models^c \theta_2$. If $S' \subseteq S$, then $S'_1 = S_1 \cap S' \subseteq S_1$ and $S'_2 = S_2 \cap S' \subseteq S_2$, hence, by induction hypothesis, $S'_1, \sigma \models^c \theta_1$ and $S'_2, \sigma \models^c \theta_2$. It follows that $(S' =) S'_1 \cup S'_2 \models^c \theta_1 \vee \theta_2$
- If $\theta = \theta_1 \wedge \theta_2$, we simply use the induction hypothesis for θ_1 and θ_2, with the same $S' \subseteq S$.
- If $\theta = \exists x.\theta_1$, then we use the induction hypothesis with the same S, S' (and a different σ). Similarly for universal quantification.

3 Computational Soundness

We assume here that, in any formula, the negations appear only in front of an atomic formula.

Theorem 1. *Let Π be a protocol, $s_1 \to \ldots \to s_m$ be a symbolic transition sequence of Π and $\mathcal{M}$ be a probabilistic polynomial time interactive Turing machine. If there is a non-negligible set of coins S such that, for any $\tau \in S$, there is a sequence of transitions $(s_0, \boldsymbol{b_0}, \boldsymbol{b'_0}, \gamma_0) \to \cdots \to (s_m, \boldsymbol{b_m}, \boldsymbol{b'_m}, \gamma_m)$ that is computationally valid w.r.t. $\mathcal{M}$, τ and γ_m is in the challenge state q_c, then for any formula θ, $\mathcal{M}, \Pi, S \models^c \theta$ implies there is a symbolic model $\mathcal{S}$ such that $s_0 \to \cdots \to s_m$ is a valid symbolic execution w.r.t. $\mathcal{S}$ and $\mathcal{S} \models \theta$.*

Proof. We assume in this proof that there are only two predicate symbols: $=$ and $\vdash$. The extension to other predicate symbols is straightforward.

For any term t with free variables $x_1, \ldots, x_n$ and machines $\mathcal{A}_{x_1}$, ..., $\mathcal{A}_{x_n}$, and any sample $\tau \in S$, let $[\![t]\!]_{\tau,\sigma}$ be the computational interpetation of t, in which each variable x_i is interepreted according to $\sigma(\tau)(x_i) = \mathcal{A}_{x_i}(b_\tau, \tau(\boldsymbol{n}))$ if b_τ is the bitstring on the output tape of γ_m, and $\boldsymbol{n}$ is the set of names in the state s_m, for the execution corresponding to τ.

Given a a decreasing chain of non-negligible sets of coins $S \supseteq S_1 \supseteq S_2 \supseteq$ ..., we define a first-order structure $\mathcal{M}_{S_1 \supseteq S_2 \supseteq \ldots}$ as follows. The domain of $\mathcal{M}_{S_1 \supseteq S_2 \supseteq \ldots}$ is the set of terms built on the function symbols, the names and the additional constants $\mathcal{A}$ for any PT machine $\mathcal{A}$. The interpretation of the predicate symbols is given, for any assignment σ of the variables $x_1, \ldots, x_n$ to machines $\mathcal{A}_{x_1}, \ldots, \mathcal{A}_{x_n}$ by:

- $\mathcal{M}_{S_1 \supseteq S_2 \supseteq \ldots}, \sigma \models t = u$ iff there is an i such that $\forall \tau \in S_i$, $[\![t]\!]_{\tau,\sigma(\tau)} = [\![u]\!]_{\tau,\sigma(\tau)}$
- $\mathcal{M}_{S_1 \supseteq S_2 \supseteq \ldots}, \sigma \models t_1, ..., t_n \vdash t$ iff there is a PT algorithm $\mathcal{A}$, an i such that $\forall \tau \in S'$, $\mathcal{A}([\![t_1]\!]_{\tau,\sigma(\tau)}, ..., [\![t_n]\!]_{\tau,\sigma(\tau)}) = [\![t]\!]_{\tau,\sigma}(\tau)$.

Let

$$\mathcal{M}_S = \mathcal{M}_{S \supseteq S \supseteq \ldots}$$

Remark 1. Notice, that the definition is such that for $S_1 \supseteq S_2 \supseteq \ldots$ and $S'_1 \supseteq S'_2 \supseteq \ldots$, if for some $m \in \mathbb{N}$, $S'_i = S_i$ for all $i > m$, then $\mathcal{M}_{S_1 \supseteq S_2 \supseteq \ldots}, \sigma \models \theta$ if and only if $\mathcal{M}_{S'_1 \supseteq S'_2 \supseteq \ldots}, \sigma \models \theta$. This is rather trivially true for θ atomic formula, and hence true for any formula.

Let θ be a formula with free variables $x_1, \ldots, x_n$ such that only atomic formulas are negated. We prove, by induction on θ that, if on a non-negligible set of coins S, $\mathcal{M}, \Pi, S, \sigma \models^c \theta$, then for any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$., there is a decreasing chain of non-negligible subsets $S'_1 \supseteq S'_2 \supseteq \ldots$. such that $S'_i \subseteq S_i$ for all $i = 1, 2, \ldots$, and for any decreasing chain of non-negligible subsets $S''_1 \supseteq S''_2 \supseteq \ldots$. with $S''_i \subseteq S'_i$ for all $i = 1, 2, \ldots$, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta$.

- Suppose $\theta \equiv t = u$. We know from Lemma 1 that $\mathcal{M}, \Pi, S, \sigma \models^c \theta$ implies $\mathcal{M}, \Pi, S', \sigma \models^c \theta$ for any subset $S' \subseteq S$. Hence, given any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$., it suffices to choose $S'_i = S_i$ for every i:
- If $\theta \equiv t \neq u$ and $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$. is any decreasing sequence of non-negligible sets, let $S'_i = S_i$ for every i. For any decreasing sequence of non-negligible sets $S''_i \subseteq S'_i$, for all i, since $S''_i, \sigma \models^c t \neq u$ by lemma 1, $\{\tau \in S''_i : [\![t]\!]_{\tau,\sigma} = [\![u]\!]_{\tau,\sigma}$ is negligible. Hence there is at least one $\tau \in S''_i$ such that $[\![t]\!]_{\tau,\sigma} \neq [\![u]\!]_{\tau,\sigma}$. Hence $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \not\models t = u$.
- For $\theta \equiv t_1, \ldots, t_n \vdash t$, again given any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$., it suffices to choose $S'_i = S_i$.
- If $\theta \equiv \hat{\phi}, u_1, \ldots, u_k \vdash t$, we may replace $\hat{\phi}$ with the frame ϕ_m of the symbolic state s_m (this is because for any $\tau \in S$, we reach the same symbolic state s_m), hence we are back to the previous case.
- If $\theta \equiv t_1, \ldots, t_n \nvdash t$, given any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$., it suffices to choose $S'_i = S_i$, as $\mathcal{M}, \Pi, S', \sigma \models^c t_1, \ldots, t_n \vdash t$ is not true on any non-negligible S'.
- If $\theta \equiv \hat{\phi}, t_1, \ldots, t_n \nvdash t$, as before, we may replace $\hat{\phi}$ with the frame in s_m and we are back to the previous case.
- If $\theta \equiv \theta_1 \vee \theta_2$, then $S, \sigma \models^c \theta$ means that there are S' and S'' such that $S' \cup S'' = S$ and $S', \sigma \models^c \theta_1$ and $S'', \sigma \models^c \theta_2$. At least one of the two sets S', S'' is non-negligible. Take any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq S_2 \supseteq \ldots$.. Then either $S' \cap S_1 \supseteq S' \cap S_2 \supseteq \ldots$. is a non-negligible chain, or $S'' \cap S_1 \supseteq S'' \cap S_2 \supseteq \ldots$. is a non-negligible chain. (Because if both are negligible from a certain point on, then there is an i such that $S' \cap S_i$ and $S'' \cap S_i$ are both negligible, but that contradicts that their union, S_i is not negligible.) Suppose the first. Notice, that the first is a chain in S'. Then, by the induction hypothesis for θ_1, there is a chain $S \supseteq S' \supseteq S'_1 \supseteq S'_2 \supseteq \ldots$ such that $S'_i \subseteq S' \cap S_i$ and for any non-negligible decreasing chain $S''_1 \supseteq S''_2 \supseteq \ldots$ with $S''_i \subseteq S'_i$, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta_1$. Then $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta$. So the same $S'_1 \supseteq S'_2 \supseteq \ldots$ works for θ.
- If $\theta \equiv \theta_1 \wedge \theta_2$, by definition, $S, \sigma \models^c \theta_1$ and $S, \sigma \models^c \theta_2$. By induction hypothesis for θ_1, given any decreasing chain of non-negligible subsets $S \supseteq S_1 \supseteq$

$S_2 \supseteq$ ...., there is a chain $S \supseteq S'_{11} \supseteq S'_{12} \supseteq$ ... with $S'_{1i} \subseteq S_i$, such that, for any non-negligible decreasing chain $S''_1 \supseteq S''_2 \supseteq$ ... with $S''_i \subseteq S'_{1i}$ for all i, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta_1$. By induction hypothesis for θ_2, there is a chain $S \supseteq S'_{21} \supseteq S'_{22} \supseteq$ ... with $S'_{2i} \subseteq S'_{1i}$ such that, for any non-negligible decreasing chain $S''_1 \supseteq S''_2 \supseteq$ ... with $S''_i \subseteq S'_{2i}$ for all i, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta_2$. Since $S'_{2i} \subseteq S'_{1i}$, by the choice of S'_{1i}, we also have that for any non-negligible decreasing chain $S''_1 \supseteq S''_2 \supseteq$ ... with $S''_i \subseteq S'_{2i}$ for all i, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta_1$. Hence, for any non-negligible decreasing chain $S''_1 \supseteq S''_2 \supseteq$ ... with $S''_i \subseteq S'_{2i}$ for all i, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \theta_1 \wedge \theta_2$. Thus, taking $S'_i = S'_{2i}$ works.

- If $\theta \equiv \exists x.\theta_1$, then there is an $\mathcal{A}_x$, for which we have that $S, \mathcal{A}_{x_1}, ..., \mathcal{A}_{x_k}, \mathcal{A}_x \models^c \theta$. By induction hypothesis, for a chain $S \supseteq S_1 \supseteq S_2 \supseteq$ ..., there is a chain $S'_1 \supseteq S'_2 \supseteq$ ... with $S'_i \subseteq S_i$, such that, for any non-negligible $S''_i \subseteq S'_i$, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_x \models \theta_1$. But then this implies $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \exists x.\theta_1$. So the same $S'_1 \supseteq S'_2 \supseteq$ ... works.
- If $\theta \equiv \forall x\theta_1$, then for all $\mathcal{A}_x$, $S, \mathcal{A}_{x_1}, ..., \mathcal{A}_{x_k}\mathcal{A}_x \models^c \theta_1$. Let's fix $S \supseteq S_1 \supseteq S_2 \supseteq$ .... Enumerate all possible algorithms for $\mathcal{A}_x$: $\mathcal{A}_1$, $\mathcal{A}_2$,... First we show that for $\mathcal{A}_1$, $S, \mathcal{A}_{x_1}, ..., \mathcal{A}_{x_k}\mathcal{A}_1 \models^c \theta_1$ holds. By induction hypothesis, there is a chain $S'_{11} \supseteq S'_{12} \supseteq S'_{13} \supseteq$ ... with $S'_{1i} \subseteq S_i$, such that, for any non-negligible $S''_1 \supseteq S''_2 \supseteq$ ..., if $S''_i \subseteq S'_{1i}$ for all i, then $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_1 \models \theta_1$. Take now $\mathcal{A}_2$. Then $S, \mathcal{A}_{x_1}, ..., \mathcal{A}_{x_k}\mathcal{A}_2 \models^c \theta_1$ holds. By the induction hypothesis, there is a chain $S'_{21} \supseteq S'_{22} \supseteq$ ... with $S'_{2i} \subseteq S'_{1i}$, such that, for any non-negligible chain $S''_1 \supseteq S''_2 \supseteq$ ... such that $S''_i \subseteq S'_{2i}$ for all i, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_2 \models \theta_1$. But, because of Remark 1, it is also true, that for the chain $S'_{11} \supseteq S'_{22} \supseteq S'_{23} \supseteq$ ..., for any non-negligible $S''_1 \supseteq S''_2 \supseteq$ ..., with $S''_1 \subseteq S'_{11}$, and $S''_i \subseteq S'_{2i}$ for $i = 2, 3...$, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_2 \models \theta_1$, as it does not matter what the first set is. Furthermore, since $S'_{2i} \subseteq S'_{1i}$ holds, we also have $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_1 \models \theta_1$. Continuing in this manner, we get a chain $S'_{11} \supseteq S'_{22} \supseteq S'_{33} \supseteq$ .... Then, take any chain $S''_1 \supseteq S''_2 \supseteq$ ..., with $S''_i \subseteq S'_{ii}$. Clearly, because of the construction, $S''_i \subseteq S'_{1i}$ also holds (as $S'_{ii} \subseteq S'_{1i}$. Hence we have $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_1 \models \theta_1$. Further, since $S''_i \subseteq S'_{2i}$ for $i = 2, 3...$, and $S''_1 \subseteq S'_{11}$, we also have $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_2 \models \theta_1$. And so on, we have for all n, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto \mathcal{A}_n \models \theta_1$. Now, if v is any term in the domain of our models, $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma, x \mapsto v \models \theta_1$. Indeed, let v' be the term v, in which any $\mathcal{A}_i$ occurring in v is replaced with a variable x_i and σ' be $x_i \mapsto \mathcal{A}_i$. The algorithm, that computes, for every $\tau \in S$, $[\![v']\!]_{\tau,\sigma'}$ can be constructed from the $\mathcal{A}_i$ and is PT. Hence there is an index n such that, for any $\tau \in S$, $\mathcal{A}_n$ outputs $[\![v]\!]_{\tau,\sigma'}$. Therefore, we also have $\mathcal{M}_{S''_1 \supseteq S''_2 \supseteq \ldots}, \sigma \models \forall x\theta_1$, and that is what we wanted to prove. □

The above result can be applied to a formula θ that is the conjunction of

- the intermediate conditions (that are part of the symbolic states) Θ
- finitely many computationally valid axioms A
- a formula that expresses the existence of an attack. NotSec.

Then it can be read as follows: if there is a computational attack, corresponding to a symbolic trace $s_1 \to \cdots \to s_m$, then this symbolic trace is valid in a model, which is also a model of A and NotSec.

Consider then a symbolic procedure, that discards only symbolic states, in which $\Theta \wedge A$ is inconsistent. Then the symbolic procedure will not miss any attack. More precisely, we get:

Theorem 2. *For a bounded number of sessions, if there is a computational attack, there is also a symbolic attack.*

In other words, if the protocol is symbolically secure, then it is also computationally secure.

It might be true for an undbounded number of sessions as well, but we need the boundedness assumption if we wish to derive the theorem from the theorem 1: The trick is, that in the bounded case, if there is a computational attack, there is also a computational attack corresponding to a fixed sequence of symbolic states. This is simply because the bounded number of sessions ensures that there are only finitely many possible sequences, and if there is a computational attack, that is, the property expressing the attack is satisfied on some non-negligible set, then it must be satisfied non-negligibly on one of the possible sequences.

4 Examples of Axioms

4.1 Examples of Axioms That Are Computationally Valid

- Increasing capabilities: $\hat{\phi} \vdash y \rightarrow \hat{\phi}, x \vdash y$
- Function of derivable items: $\hat{\phi} \vdash t_1 \wedge \hat{\phi} \vdash t_2 \wedge ... \wedge \hat{\phi} \vdash t_n \rightarrow \hat{\phi} \vdash f(t_1, t_2, ..., t_n)$
- Self derivability: $\hat{\phi}, t \vdash t$

The validity of these axioms is straightforward. We also include the following:

No telepathy: $\mathsf{fresh}(x, \hat{\phi}) \rightarrow \hat{\phi} \nvdash x$

whose computational soundness follows from the polynomial bound on the machines that interpret the deducibility relation on the one hand and the exponential number of interpretations of any names, on the other hand.

4.2 Secrecy Axiom

The intuitive meaning of the following axiom is that the adversary cannot derive the plaintext of a freshly generated encryption, unless its decryption key has been sent out, or the plaintext could be derived earlier.

Proposition 1. *If the encryption scheme is IND-CCA2, then the following axiom*

$$\theta = \forall_{tKR}\Big(\mathsf{RandGen}(K) \wedge \mathsf{fresh}(R, \hat{\phi}) \wedge \hat{\phi}, \{t\}^R_{eK} \vdash t \longrightarrow dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t\Big)$$

is computationally valid.

Proof. Suppose that it is not computationally valid. That is, there is a computational structure $(\mathcal{M}, \Pi, S)$, with $\mathcal{M}, \Pi, S \not\models \theta$. There are PPT machines $\mathcal{A} = (\mathcal{A}_t, \mathcal{A}_K, \mathcal{A}_R)$ such that $\mathcal{M}, \Pi, S, \mathcal{A} \not\models \mathsf{fresh}(R, \hat{\phi}) \wedge \hat{\phi}, \{t\}^R_{eK} \vdash t \longrightarrow dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t$. Therefore, there is a $S_1 \subseteq S$ non-negligible such that $\mathcal{M}, \Pi, S_1, \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi}) \wedge \hat{\phi}, \{t\}^R_{eK} \vdash t$ and $\mathcal{M}, \Pi, S_1, \mathcal{A} \not\models dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t$. We claim that the second implies that there is a non-negligible subset S_2 of S_1 such that $\mathcal{M}, \Pi, S_2, \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$ and $\mathcal{M}, \Pi, S_2, \mathcal{A} \not\models \hat{\phi} \vdash t$. To see this, consider the following:

- Take $S_2 = S_1 \backslash \{\tau \mid$ the computation of $\mathcal{A}$ on τ yields a state q such that $q \models dK \sqsubseteq \hat{\phi}\}$. Clearly, $\mathcal{M}, \Pi, S_2, \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$, and $\mathcal{M}, \Pi, S_1 \setminus S_2, \mathcal{A} \models dK \sqsubseteq \hat{\phi}$
- Since $\mathcal{M}, \Pi, S_1 \setminus S_2, \mathcal{A} \models dK \sqsubseteq \hat{\phi}$, we have $\mathcal{M}, \Pi, S_2, \mathcal{A} \not\models \hat{\phi} \vdash t$, because otherwise we would have $\mathcal{M}, \Pi, S_1, \mathcal{A} \models dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t$ contradicting $\mathcal{M}, \Pi, S_1, \mathcal{A} \not\models dK \sqsubseteq \hat{\phi} \vee \hat{\phi} \vdash t$.

Since $\mathcal{M}, \Pi, S_2, \mathcal{A} \not\models \hat{\phi} \vdash t$, by the definition of the computational semantics of the derivability predicate, there is a subset S_4 of S_2 such that on all subsets of S_4, there is no PT algorithm that computes the interpretation of t from the computational frame. Then it is straightforward to check that $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \neg(\hat{\phi} \vdash t)$:

- Suppose it is not true, that is, $\mathcal{M}, \Pi, S_4, \mathcal{A} \not\models \neg(\hat{\phi} \vdash t)$.
- Then there is an $S_5 \subseteq S_4$ such that $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \hat{\phi} \vdash t$.
- That implies that S_5 has a subset on which there is an algorithm that computes the interpretation t from the computational frame, a contradiction.

Since $S_4 \subseteq S_2$, we also have that $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$, and since $S_4 \subseteq S_1$, we also have $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi}) \wedge \hat{\phi}, \{t\}^R_{eK} \vdash t$. That is, $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \hat{\phi}, \{t\}^R_{eK} \vdash t$ and $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi})$ and $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$ and $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \neg(\hat{\phi} \vdash t)$. We have to create an adversary $\mathcal{A}_{\mathrm{CCA2}}$ that wins the CCA2 game. Let $x = \{t\}^R_{eK}$.

Since $\mathcal{M}, \Pi, S_4, \mathcal{A} \models \hat{\phi}, \{t\}^R_{eK} \vdash t$ holds, there is an $S_5 \subseteq S_4$ and an algorithm $\mathcal{C}$ that computes the interpretation of t from the interpretation of $\hat{\phi}$ and $\{t\}^R_{eK}$ on S_5. Clearly, $\mathcal{M}, \Pi, S_5, \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi})$ and $\mathcal{M}, \Pi, S_5, \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$ and $\mathcal{M}, \Pi, S_5, \mathcal{A} \models \neg(\hat{\phi} \vdash t)$. It may be the case that the S_5 we have chosen depends on evaluations of τ that are determined after $\mathcal{M}$ reaches the challenge state q_c. However, clearly, if we include all possible future evaluations, the set that we receive this way, S' will still be such that there is an algorithm $\mathcal{C}$ that computes the interpretation of t from the frame at the challenge state q_c and $\{t\}^R_{eK}$ on S'. Moreover, it is easy to see that $\mathcal{M}, \Pi, S', \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi})$ and $\mathcal{M}, \Pi, S', \mathcal{A} \models \neg(dK \sqsubseteq \hat{\phi})$ and $\mathcal{M}, \Pi, S', \mathcal{A} \models \neg(\hat{\phi} \vdash t)$ because these are properties that depend only on conditions in the challenge stated, and not later ones.

Since $\mathcal{M}, \Pi, S', \mathcal{A} \models dK \not\sqsubseteq \hat{\phi}$, the decryption key has never been sent.

We show that we can construct an algorithm $\mathcal{A}_{\mathrm{CCA2}}$ that breaks CCA2 security.

Let $\mathcal{A}_\Pi$ mean the protocol adversary.

- $\mathcal{A}_{\mathrm{CCA2}}$ generates computational keys that $\mathcal{A}_\Pi$ uses, except for the one corresponding to K.

- The encryption oracle generates a random bit b.
- The encryption oracle generates a computational key and publishes its public part. $\mathcal{A}_{\mathrm{CCA2}}$ encrypts with this key for encryptions with K, except for t.
- $\mathcal{A}_{\mathrm{CCA2}}$ simulates both the agents and $\mathcal{A}_{\Pi}$: It computes all messages that the agents output according to their algorithm, and computes all messages that $\mathcal{A}_{\Pi}$ outputs according to its algorithm. This way it builds up ϕ and the bit strings corresponding to them as well as the equations.
- Whenever a decryption with dK has to be computed, there are two possibilities:
  - If the ciphertext was created by $\mathcal{A}_{\mathrm{CCA2}}$ using the encryption algorithm, then it knows the plaintext, so it can use it without decryption.
  - If the ciphertext was created in some other way, the decryption oracle is used. This can be freely done until x occurs.
- When $\mathcal{A}$ reaches the challenge state q_c, using $\mathcal{A}_t$, $\mathcal{A}_{\mathrm{CCA2}}$ computes the bit string for t, and submits it to the encryption oracle as well as a random bit string that has the same length as the plaintext.
- According to our definition of satisfaction the computation by $\mathcal{C}$ is based on the frame at the challenge state. We had $\mathcal{M}, \Pi, S', \mathcal{A} \models \mathsf{fresh}(R, \hat{\phi})$, which means that R is independent of the items in ϕ. Further, since we included all future random choices in S', R is also independent of S'. Hence having it encrypted by the encryption oracle will not lose any information as long as the oracle encrypts the correct bit.
- The encryption oracle encrypts the interpretation of t if $b = 0$, and encrypts the random bit string if $b = 1$. It gives the result c back to $\mathcal{A}_{\mathrm{CCA2}}$.
- Run $\mathcal{C}$ on the bit string c returned by the oracle and on the bit strings of $\phi_{\mathbf{n}}$.
- If
  - $\mathcal{A}_{\mathrm{CCA2}}$ receives the value for t back using c and if the execution is in S', then $\mathcal{A}_{\mathrm{CCA2}}$ returns $b_{\mathcal{A}_{\mathrm{CCA2}}} = 0$.
  - Otherwise $\mathcal{A}_{\mathrm{CCA2}}$ throws a fair coin and stores $b_{\mathcal{A}_{\mathrm{CCA2}}} = 0$ or $b_{\mathcal{A}_{\mathrm{CCA2}}} = 1$ with probability $1/2$.
- We have $\mathbf{Prob}\{b_{\mathcal{A}_{\mathrm{CCA2}}} = b \mid S' \wedge b = 0\}$ (the conditional probability of $b_{\mathcal{A}_{\mathrm{CCA2}}} = b$ given S' and $b = 0$) is negligibly different from 1 because in this case the oracle encrypts the correct string, and $\mathcal{C}$'s computations are employed on the correct bit string, and so it gives the interpretation of t. Note, we also use here that S' and the interpretation of R do not correlate.
- On the other hand, observe that $\mathbf{Prob}\{b_{\mathcal{A}_{\mathrm{CCA2}}} = b \mid S' \wedge b = 1\} - 1/2$ is negligible. The reason is that when $b = 1$, the encryption oracle computes something that has nothing to do with the protocol and t. So the probability of computing t with or without the encryption in this case, is the same. But, remember, we had that $\mathcal{M}, \Pi, S', \mathcal{A} \models \hat{\phi} \not\vdash t$. This means that t cannot be computed without the encryption anywhere and therefore the adversary's computation on the fake encryption cannot give good result by more than negligible probability. So the adversary will end up throwing a coin in this case.
- Putting the previous two points together, we have $\mathbf{Prob}\{b_{\mathcal{A}_{\mathrm{CCA2}}} = b \mid S'\} - \frac{1}{2}$ is non-negligible. Then, since outside S', $\mathcal{A}_{\mathrm{CCA2}}$ thows a coin, $\mathbf{Prob}\{b_{\mathcal{A}_{\mathrm{CCA2}}} = b\} - \frac{1}{2}$ is non-negligible, which means CCA2 security is broken. □

5 Conclusions

We have shown a technique to define symbolic adversaries that are at least as strong as computational adversaries. The basic idea is that, instead of listing all manipulations the symbolic adversary is allowed to do, we allow the symbolic adversary to do anything unless it contradicts some axioms, which are derived from the limitations of the computational adversary. In a rather involved theorem, we showed that at least when only bounded number of protocol sessions are allowed, to any computational attack there is a corresponding symbolic attack. Further, we have shown a few axioms that arise from the limitations of computational adversaries, and which are to limit the symbolic adversary. Besides some rather trivially valid axioms, we showed the validity of a "secrecy axiom", that relies on IND-CCA2 security.

From our method, we can derive a verification procedure, simulating the (symbolic) protocol rules, and checking at each computation step the consistency of the formulas expressing that transitions are enabled, together with the axioms and the negation of the security properties. In order to automate this process we mainly need a (hopefully efficient) procedure checking the consistency of such a set of constrained formulas. This is future work. We are however optimistic, because the examples of axioms that we considered yield a saturated set of constrained formulas (as defined in [22]). On the other hand, as shown in [9], the consistency of ground clauses, together with a saturated set of clauses, can be performed in polynomial time.

We carried out a proof of a two sessions NSL, showing what are the minimal assumptions that guarantee its correctness, but we need to design an automated tool, in order to carry out further experiments. Also extensions of the results to indistinguishability properties could be investigated.

References

1. Abadi, M., Rogaway, P.: Reconciling Two Views of Cryptography: the Computational Soundness of Formal Encryption. In: Watanabe, O., Hagiya, M., Ito, T., van Leeuwen, J., Mosses, P.D. (eds.) TCS 2000. LNCS, vol. 1872, pp. 3–22. Springer, Heidelberg (2000)
2. Abadi, M., Blanchet, B., Comon-Lundh, H.: Models and Proofs of Protocol Security: A Progress Report. In: Bouajjani, A., Maler, O. (eds.) CAV 2009. LNCS, vol. 5643, pp. 35–49. Springer, Heidelberg (2009)
3. Backes, M., Pfitzmann, B.: Symmetric encryption in a simulatable dolev-yao style cryptographic library. In: Proc. IEEE Computer Security Foundations Workshop (2004)
4. Backes, M., Pfitzmann, B.: Limits of the Cryptographic Realization of Dolev-Yao-Style XOR. In: di Vimercati, S.d.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 178–196. Springer, Heidelberg (2005)
5. Backes, M., Pfitzmann, B., Waidner, M.: A composable cryptographic library with nested operations. In: Proc. 10th ACM Concerence on Computer and Communications Security, CCS 2003 (2003)
6. Backes, M., Pfitzmann, B., Waidner, M.: The reactive simulatability (rsim) framework for asynchronous systems. Information and Computation 205(12) (2007)
7. Backes, M., Hofheinz, D., Unruh, D.: Cosp: A general framework for computational soundness proofs. In: ACM CCS 2009, pp. 66–78 (November 2009); Preprint on IACR ePrint 2009/080

8. Bana, G., Hasebe, K., Okada, M.: Secrecy-oriented first-order logical analysis of cryptographic protocols (2010), http://eprint.iacr.org/2010/080
9. Basin, D., Ganzinger, H.: Automated complexity analysis based on ordered resolution. Journal of the Association of Computing Machinery 48(1), 70–109 (2001)
10. Blanchet, B.: An automatic security protocol verifier based on resolution theorem proving (invited tutorial). In: 20th International Conference on Automated Deduction (CADE-20), Tallinn, Estonia (July 2005)
11. Blanchet, B.: A computationally sound mechanized prover for security protocols. IEEE Transactions on Dependable and Secure Computing 5(4), 193–207 (2008); Special issue IEEE Symposium on Security and Privacy (2006)
12. Canetti, R., Rabin, T.: Universal Composition with Joint State. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 265–281. Springer, Heidelberg (2003)
13. Comon-Lundh, H., Cortier, V.: Computational soundness of observational equivalence. In: Proc. ACM Conf. Computer and Communication Security, CCS (2008)
14. Comon-Lundh, H., Cortier, V.: How to prove security of communication protocols? a discussion on the soundness of formal models w.r.t. computational ones. In: Dürr, C., Schwentick, T. (eds.) Proceedings of the 28th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2011), Dortmund, Germany. Leibniz International Proceedings in Informatics, vol. 9, pp. 29–44. Leibniz-Zentrum für Informatik (March 2011)
15. Cortier, V., Warinschi, B.: Computationally Sound, Automated Proofs for Security Protocols. In: Sagiv, M. (ed.) ESOP 2005. LNCS, vol. 3444, pp. 157–171. Springer, Heidelberg (2005)
16. Datta, A., Derek, A., Mitchell, J.C., Turuani, M.: Probabilistic Polynomial-Time Semantics for a Protocol Security Logic. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 16–29. Springer, Heidelberg (2005)
17. Armando, A., Basin, D., Boichut, Y., Chevalier, Y., Compagna, L., Cuellar, J., Drielsma, P.H., Heám, P.C., Kouchnarenko, O., Mantovani, J., Mödersheim, S., von Oheimb, D., Rusinowitch, M., Santiago, J., Turuani, M., Viganò, L., Vigneron, L.: The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In: Etessami, K., Rajamani, S.K. (eds.) CAV 2005. LNCS, vol. 3576, pp. 281–285. Springer, Heidelberg (2005)
18. Ganzinger, H., Nieuwenhuis, R.: Constraints and Theorem Proving. In: Comon, H., Marché, C., Treinen, R. (eds.) CCL 1999. LNCS, vol. 2002, pp. 159–201. Springer, Heidelberg (2001)
19. Lifschitz, V.: Closed-world databases and circumscription. Artif. Intell. 27(2), 229–235 (1985)
20. Lowe, G.: Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR. In: Margaria, T., Steffen, B. (eds.) TACAS 1996. LNCS, vol. 1055, pp. 147–166. Springer, Heidelberg (1996)
21. Millen, J., Shmatikov, V.: Constraint solving for bounded-process cryptographic protocol analysis. In: Proc. 8th ACM Conference on Computer and Communications Security (2001)
22. Nieuwenhuis, R., Rubio, A.: Paramodulation-based theorem proving. In: Handbook of Automated Reasoning, pp. 371–443. Elsevier and MIT Press (2001)
23. Ryan, P., Schneider, S., Goldsmith, M., Lowe, G., Roscoe, B.: The Modelling and Analysis of Security Protocols. Addison Wesley (2000)
24. Unruh, D.: Computational soundness of hash functions. Presented at the 6th Workshop on Formal and Computational Cryptography (FCC) (July 2010)
25. Unruh, D.: The impossibility of computationally sound xor (July 2010); Preprint on IACR ePrint 2010/389
26. Warinschi, B.: A computational analysis of the needham-schroeder protocol. In: 16th Computer Security Foundation Workshop (CSFW), pp. 248–262. IEEE (2003)