Workshop on Logical Aspects of Cryptographics 2001 Preliminary Version

Models for an Adversary-Centric Protocol Logic

Peter Selinger

Department of Mathematics and Statistics
University of Ottawa
Ottawa, Canada

Abstract

In this paper, we propose an adversary-centric, logical framework for formalizing cryptographic protocols. The formalism is inspired by the work of Compton and Dexter and of Cervesato et al., but we do not focus on proof search, but instead on logical validity. A novel contribution of this paper is a technique for giving very short proofs of protocol correctness through models of first-order logic.

1 Introduction

Logic-based approaches to the analysis of cryptographic protocols can be roughly divided into protocol-centric and adversary-centric approaches. In the *protocol-centric* approaches, one devises a formal system such that a derivation of a certain formula in the system corresponds to a proof of correctness of the protocol. Conversely, the absence of a derivation indicates the potential for an attack. In particular, if a model can be given which falsifies the target formula, then such a model corresponds to a potential attack. This observation (in principle) provides the basis for the application of traditional model-checking techniques to cryptographic protocol analysis. Protocol-centric approaches include the original BAN logic by Burrows et al. [1], as well as many systems directly or indirectly derived from it.

In the *adversary-centric* approaches, the situation is reversed. Here, a logical system is given such that derivations correspond to possible *attacks* on a protocol. The correctness of a protocol is thus witnessed by the *absence* of a derivation of a certain formula. Such approaches have been mostly developed in connection with *proof search*; the idea is to use standard proof search techniques to search for attacks on protocols. This is the approach followed by Compton and Dexter [3] and, in different form, by Cervesato et al. [2].

[1] This research was done while the author was at Stanford University.

This is a preliminary version. The final version will be published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

The purpose of the present paper is to take adversary-centric protocol logics out of the context of proof search, and to view them in a more general logical context. This allows us to move beyond the world of Horn formulas common in logic programming, and we obtain an elegant encoding of protocols as certain first-order formulas. Based on this encoding, we explore the idea of using *models* to demonstrate the correctness of a protocol. Note that in this framework, a single countermodel suffices to prove the absence of an attack, and thus the correctness of a protocol. This contrasts with the usual model checking techniques, where each model corresponds to a particular *run* of the protocol, and thus an exploration of all possible models is needed in order to prove correctness.

All logic-based approaches to protocol analysis, whether they are protocol-centric or adversary-centric, make certain assumptions about the properties of the cryptographic primitives from which the protocols are built, and about the types of attacks that are possible. Here, we assume a standard Dolev-Yao "black-box" model of cryptography, whereby messages are modeled as terms, and cryptographic operations are modeled in an idealized, "perfect" way [4]. We assume the presence of an intruder who may actively participate in the protocol by reading, altering and/or blocking any messages on the network, inserting new messages into the network, and remembering and analyzing any data seen. In particular, the intruder may apply a known encryption or decryption key to a known message, decompose messages into parts and re-compose them, and generate fresh data as needed.

2 Logical encoding of protocols

2.1 Cryptographic primitives and the "black-box" model

Following Dolev and Yao, we model messages as terms that are built from variables and some or all of the following operations. The language may vary slightly depending on the protocol modeled.

$$\begin{aligned} m, k ::= x \mid [m]_k \mid \{m\}_k \mid k^- \mid (m_1, \ldots, m_n) \mid \langle m \rangle_k \mid F(m) \\ \mid \bar{A} \mid K_A \mid K_{A,B} \mid s_{A,B}. \end{aligned}$$

Here A, B, C range over a sort of *principals*, while m, k etc. range over *data*, which includes keys, messages, identifiers, etc. We do not introduce separate sorts for particular kinds of data such as keys; thus, any piece of data could be potentially used as an encryption key, or in any other way. The intended interpretation of these terms is as follows: $[m]_k$ is the symmetric key encryption of m with key k. $\{m\}_k$ is the public key encryption of m with key k. k^- is the private key corresponding to public key k. $(m_1, \ldots, m_n)$ is a tuple of messages. Tuples should always be uniquely decomposable; in particular, we make no assumption that the tupling operation is associative. We also write $[m, m']_k$ as a shortcut for $[(m, m')]_k$, etc. $\langle m\rangle_k$ is a keyed hash of m under key k. $F(m)$ is a "handshake" function applied to m. A handshake function is a trivially reversible operation, as used in some protocols; for instance, it is often taken to be $m+1$. $\bar{A}$ is the name of principal A. The last three term constructors are highly protocol-specific. K_A stands for A's long-term public key, in protocols that assume that such keys are available. Similarly, $K_{A,B}$ stands for a long-term shared key between A and B, while $s_{A,B}$ stands for any other long-term secret shared between A and B.

Depending on the protocol, there may also be one or more constant symbols denoting certain principals; for instance, the constant S is often used to denote a principal who acts as a server. In addition, we always have a constant symbol I to denote the intruder, who is also a principal.

The properties of the various term constructors are axiomatized in Table 1. These axioms are expressed in two-sorted first-order logic, where we have omitted outermost universal quantifiers for brevity. An important feature of these axioms is that they do not involve equations, but rather, they are expressed in terms of a single, unary predicate symbol $\boldsymbol{K}$. This predicate symbol is intended to capture what the intruder can know. Thus we read $\boldsymbol{K}\,m$ as "m is knowable" by the intruder. Note that, unlike in "logics of knowledge", $\boldsymbol{K}$ is a predicate, not a modality; it is applied to terms, not to formulas. There is also one predicate symbol on principals: $\boldsymbol{H}\,A$ is intended to mean "A is honest." The honesty predicate is primarily used in the formulation of correctness properties (such as, "the intruder cannot learn any honest principals' secret").

Having stated the properties of the cryptographic primitives, we can now be more precise what we mean by "black-box" cryptography.

Assumption (Black-box assumption). The axioms in Table 1 are the *only* way in which a passive intruder can infer new knowledge from known data.

Obviously, this is a rather strong assumption to make in practice, since in real-world cryptography, it is often possible to infer partial or statistical information about the content of a message, without necessarily understanding the entire message. However, the black-box assumption is essential to our model, and in fact to most other logic-based models of cryptographic protocols.

In addition to the general-purpose axioms shown in Table 1, there will also be protocol-specific axioms. The intuition is that the general-purpose axioms capture all the ways in which a *passive* intruder can infer knowledge, namely by looking at data and analyzing it. The protocol-specific axioms, discussed in the next section, capture the additional ways in which an *active* intruder can infer knowledge by interacting with a particular protocol.

$\boldsymbol{K}\,k \to (\boldsymbol{K}\,m \leftrightarrow \boldsymbol{K}[m]_k)$	symmetric key
$\boldsymbol{K}\,k \wedge \boldsymbol{K}\,m \to \boldsymbol{K}\{m\}_k$	public key encryption
$\boldsymbol{K}\{m\}_k \wedge \boldsymbol{K}\,k^- \to \boldsymbol{K}\,m$	public key decryption
$\boldsymbol{K}(m_1,\ldots,m_n) \leftrightarrow (\boldsymbol{K}\,m_1 \wedge \ldots \wedge \boldsymbol{K}\,m_n)$	tuples
$\boldsymbol{K}\,m \wedge \boldsymbol{K}\,k \to \boldsymbol{K}\langle m\rangle_k$	hash
$\boldsymbol{K}\,m \leftrightarrow \boldsymbol{K}\,F(m)$	handshake
$\boldsymbol{K}\,k^- \to \boldsymbol{K}\,k$	key generation
$\boldsymbol{K}\,\bar{A}$	all identities are known
$\boldsymbol{K}\,K_A$	all public keys are known
$\boldsymbol{K}\,K_I^-$	intruder has a private key
$\boldsymbol{K}\,K_{A,I} \wedge \boldsymbol{K}\,K_{I,A}$	intruder shares a key with each A
$\boldsymbol{K}\,s_{A,I} \wedge \boldsymbol{K}\,s_{I,A}$	intruder shares a secret with each A
$\exists A.\,\boldsymbol{H}\,A$	there exists an honest principal
$\neg\,\boldsymbol{H}\,I$	the intruder is not honest

Table 1
Properties of the cryptographic primitives

2.2 Protocol-specific axioms

The encoding of specific protocols is best illustrated in an example. Consider the following public key version of the Needham-Schroeder protocol. The Needham-Schroeder protocol is an authentication protocol which is written in traditional notation as follows:

$$(1)\ A \to B : \{\bar{A}, N_a\}_{K_B}$$
$$(2)\ B \to A : \{N_a, M_b\}_{K_A}$$
$$(3)\ A \to B : \{M_b\}_{K_B}$$

Here A, B are two principals whose goal it is to establish their mutual identity while trading a pair of secret nonces N_a and M_b. (Nonces are unique identifiers generated on the fly and never assumed to be used more than once). Each principal A has a given long-term public key K_A. N_a is a nonce generated by A, and M_b is a nonce generated by B.

The informal requirement of this protocol is that at the end of their session, each of A and B should be convinced of the other's authenticity, and moreover, that N_a and M_b are shared secrets between A and B. (Here "shared secret" means that the information should not be known to anybody besides A and B unless it was intentionally disclosed by one of them).

Authentication properties are difficult to formalize directly, since they involve reasoning about principals' beliefs. We avoid this issue by using a standard trick for reducing an *authentication* property to a *secrecy* property. To this end, we assume that any two principals A and B share some a priori secret $s_{A,B}$, and we add two more steps to the protocol:

$$(4)\ A \to B : [s_{A,B}]_{N_a}, [s_{A,B}]_{M_b}$$
$$(5)\ B \to A : [s_{A,B}]_{N_a}, [s_{A,B}]_{M_b}$$

The additional steps (4) and (5) can be thought of as modeling a "payload" communication which follows the initial authentication (1)-(3). The idea here is that after A and B are convinced of each other's identity, they "act" on their belief by revealing a secret to each other. The requirement of the Needham-Schroeder protocol can now be expressed (still informally) in terms of a secrecy property: the intruder should not be able to learn $s_{A,B}$, for any honest principals A, B. The same trick for expressing an authentication property as a secrecy property was used in [3].

Although it is not clear from the above notation, we intend that steps (4) and (5) do not causally depend on each other; in particular, B will send message (5) immediately upon completion of step (3).

To formalize this protocol in our first-order logic, we must translate it into axioms that describe all the possible ways in which an intruder may gain knowledge by interacting with this protocol. While it seems, at first glance, that some information about the protocol might be lost during this translation, in fact the opposite is the case: as we will see, our formalism describes the protocol in rather more detail than the above "traditional" notation.

Each protocol has a set of "roles" that can be filled by principals. For instance, the above public key Needham-Schroeder protocol has two roles: the "initiator" (A) and the "responder" (B). Each role specifies how a principal reacts to incoming messages, and thus gives rise to a potential source of information for the intruder. Thus, for each role in the protocol, we obtain one axiom describing how the intruder can gain knowledge by interacting with that role. Here are the axioms for the public key Needham-Schroeder protocol:

$$\Phi_i : \forall A, B\, \exists n\, (\boldsymbol{K}\{\bar{A}, n\}_{K_B} \wedge \forall m\, (\boldsymbol{K}\{n, m\}_{K_A} \to (\boldsymbol{K}\{m\}_{K_B} \wedge (\boldsymbol{K}[s_{A,B}]_n \wedge \boldsymbol{K}[s_{A,B}]_m))))$$
$$\Phi_r : \forall A, B\, \forall n\, (\boldsymbol{K}\{\bar{A}, n\}_{K_B} \to \exists m\, (\boldsymbol{K}\{n, m\}_{K_A} \wedge (\boldsymbol{K}\{m\}_{K_B} \to (\boldsymbol{K}[s_{A,B}]_n \wedge \boldsymbol{K}[s_{A,B}]_m))))$$

The axiom Φ_i corresponds to the initiator role, and Φ_r corresponds to the responder role.

2.3 Discussion of the axioms

The first thing one might notice about the axioms Φ_i and Φ_r is the usage of existential quantifiers to model the creation of nonces; this follows the same ideas as in [2]. For instance, we can interpret Φ_r as follows: if the intruder knows any message of the form $\{\bar{A}, n\}_{K_B}$, then he can learn a message of the form $\{n, m\}_{K_A}$, for some m. Namely, he achieves this by sending the message $\{\bar{A}, n\}_{K_B}$ to B, who will then respond in the way prescribed by step (2) of the protocol. Moreover, if the intruder then knows $\{m\}_{K_B}$, for the *same* m that was used in the previous step, then he can learn $[s_{A,B}]_n$ and $[s_{A,B}]_m$.

This demonstrates how, by using nested formulas with quantifiers, we can model the fact that principals have a *memory*: the responder will check that the m received in step (3) was actually the same as the m she previously generated.

Now consider Φ_i. The encoding works much the same way as that of Φ_r, but notice that Φ_i has an existential quantifier at the top level. This means, an intruder can learn $\{\bar{A}, n\}_{K_B}$, for some n, without any prior knowledge. Thus, we have modeled the fact that any principal A might spontaneously assume the role of initiator and attempt to communicate with any B. In effect, we have given the intruder the power to "coax" A into initiating communication with B.

Another thing one might notice about the formulas Φ_i and Φ_r is their near-perfect symmetry. Existential quantifiers line up with universals, and conjunctions with implications. This is not a coincidence; it reflects the fact that an intended run of the protocol produces the intended result. Indeed, the reader may easily verify that the two formulas Φ_i and Φ_r logically imply $\exists k. \boldsymbol{K}[s_{A,B}]_k$. The proof corresponds precisely to an intended run of the protocol, which results in messages of the form $[s_{A,B}]_k$ being sent.

Another interesting property of Φ_i and Φ_r is that we can verify that principals actually have the knowledge required to run this protocol. While this is a liveness property and not a security property, it is certainly important that a protocol should not be "over-specified" by relying on the principals to use information that they do not have access to. The reader may verify that in the presence of the axioms from Table 1, $\Phi_i(A, B)$ is logically derivable from $\boldsymbol{K}\, K_A^-$ and $\boldsymbol{K}\, s_{A,B}$, while $\Phi_r(A, B)$ is logically derivable from $\boldsymbol{K}\, K_B^-$ and $\boldsymbol{K}\, s_{A,B}$. Thus, any principal with knowledge of a private key can participate in this protocol.

2.4 Attacks as proofs

Having given the axioms by which an intruder may infer knowledge from interacting with a protocol, we can now state the intended correctness property of the protocol formally. Recall that the informal requirement was that the intruder should not be able to learn the secret $s_{A,B}$ for any honest principals A and B. Let Ξ be the set of axioms from Table 1. Let Ψ be the following formula, which expresses a violation of secrecy:

$$\Psi: \quad \exists A, B.\, \boldsymbol{H}\, A \wedge \boldsymbol{H}\, B \wedge \boldsymbol{K}\, s_{A,B}.$$

Then the formal correctness requirement is

$$\Xi, \Phi_i, \Phi_r \not\vdash \Psi,$$

i.e., Ψ is not derivable from Ξ, Φ_i, and Φ_r.

It is well-known that the public key Needham-Schroeder protocol does not satisfy this requirement. Indeed, the intruder can learn $s_{A,B}$ for all A and B. The attack, due to Lowe [5], assumes that the intruder I can pose as an honest principal with its own public key K_I, and that A initiates communication with I. In addition, of course, the intruder can pose as another principal.

In our framework, Lowe's attack translates into a proof of Ψ as follows. Let A and B be some honest principals. From $\Phi_i(A, I)$, it follows that there exists some n such that $\boldsymbol{K}\{\bar{A}, n\}_{K_I}$ and

$$\forall m\, (\boldsymbol{K}\{n, m\}_{K_A} \to (\boldsymbol{K}\{m\}_{K_I} \wedge (\boldsymbol{K}[s_{A,I}]_n \wedge \boldsymbol{K}[s_{A,I}]_m))). \tag{1}$$

From $\boldsymbol{K}\, K_I^-$, we get $\boldsymbol{K}(\bar{A}, n)$, thus also $\boldsymbol{K}\, n$. From $\boldsymbol{K}\, K_B$, we get $\boldsymbol{K}\{\bar{A}, n\}_{K_B}$. From $\Phi_r(A, B)$, it follows that there exists some m such that $\boldsymbol{K}\{n, m\}_{K_A}$ and

$$\boldsymbol{K}\{m\}_{K_B} \to (\boldsymbol{K}[s_{A,B}]_n \wedge \boldsymbol{K}[s_{A,B}]_m). \tag{2}$$

From (1), we get $\boldsymbol{K}\{m\}_{K_I}$, from which we conclude $\boldsymbol{K}\, m$ and $\boldsymbol{K}\{m\}_{K_B}$ by using $\boldsymbol{K}\, K_I^-$ and $\boldsymbol{K}\, K_B$ as before. Finally, from (2), we get $\boldsymbol{K}[s_{A,B}]_n$ and $\boldsymbol{K}[s_{A,B}]_m$. Then $\boldsymbol{K}\, n$ and $\boldsymbol{K}\, m$ imply $\boldsymbol{K}\, s_{A,B}$, as desired.

For reference, here is Lowe's attack in traditional notation:

$$
\begin{array}{ll}
(1') & A \to I : \{\bar{A}, N_a\}_{K_I} \\
(1'') & I_A \to B : \{\bar{A}, N_a\}_{K_B} \\
(2'') & B \to I_A : \{N_a, M_b\}_{K_A} \\
(2') & I \to A : \{N_a, M_b\}_{K_A} \\
(3') & A \to I : \{M_b\}_{K_I} \\
(3'') & I_A \to B : \{M_b\}_{K_B} \\
(5'') & B \to I_A : [s_{A,B}]_{N_a}, [s_{A,B}]_{M_b}
\end{array}
$$

Here I_A is the standard notation for "I pretending to be A". Note that our derivation of Ψ from Φ_i, Φ_r, and the other axioms indeed corresponds very closely to the way the intruder gains knowledge in Lowe's attack. In fact, the derivation gives rather more detail about how the attack is carried out.

2.5 Correctness via a model

As Lowe also pointed out, the public-key Needham-Schroeder protocol can easily be fixed by including B's name in the second message. We can formalize the fixed protocol as follows:

$$\Phi_i' : \forall A, B\, \exists n\, (\boldsymbol{K}\{\bar{A}, n\}_{K_B} \wedge \forall m\, (\boldsymbol{K}\{n, m, \bar{B}\}_{K_A} \to (\boldsymbol{K}\{m\}_{K_B} \wedge (\boldsymbol{K}[s_{A,B}]_n \wedge \boldsymbol{K}[s_{A,B}]_m))))$$

$$\Phi_r' : \forall A, B\, \forall n\, (\boldsymbol{K}\{\bar{A}, n\}_{K_B} \to \exists m\, (\boldsymbol{K}\{n, m, \bar{B}\}_{K_A} \wedge (\boldsymbol{K}\{m\}_{K_B} \to (\boldsymbol{K}[s_{A,B}]_n \wedge \boldsymbol{K}[s_{A,B}]_m))))$$

To prove the correctness of the fixed protocol, we must prove that the formula Ψ is *not* derivable from Ξ, Φ_i', and Φ_r'. Since this protocol does not use a shared private key, hash functions, or handshake functions, we may omit those features from the signature of the language for the purpose of the present discussion.

To prove that Ψ cannot be derived from the axioms, it suffices to give a first-order model which validates the axioms, but not Ψ. One such model is shown in Table 2. Note that this is a model in the standard sense of models of first-order logic. The two sorts of principals and data are interpreted by sets $\mathbb{P}$ and $\mathbb{D}$, respectively. The unary predicates $\boldsymbol{H}$ and $\boldsymbol{K}$ are interpreted as subsets of $\mathbb{P}$ and $\mathbb{D}$, respectively, and the unary and binary operations are interpreted as shown. The ternary tripling operation (a, b, c) is defined in terms of the pairing operation as $((a, b), c)$.

It is easy, though tedious, to verify that the resulting model indeed validates all the given axioms, including Φ_i' and Φ_r'. On the other hand, it does not validate Ψ. Thus, the model proves the correctness of the corrected public key Needham-Schroeder protocol, with respect to our assumptions about black-box cryptography.

2.6 Discussion of the model

It seems counterintuitive at first that a single model, let alone such a trivial one with five elements, is supposed to prove the correctness of a protocol. To help the intuition, imagine a universe of all possible data. Imagine that this universe was partitioned into two sets: the top-secret data, and the data which is publicly known. Each of these sets might be further partitioned: for instance the top-secret data might be further divided into data whose encryption under k is known or not known, for a certain key k. Proceeding in this way, we divide the set of all possible data into equivalence classes. Proving the correctness of a certain protocol might now be likened to defining a line between known and unknown information, and proving that the protocol does not, in an appropriate sense, cross the line. The above model, then, provides a particular way of partitioning the universe into five equivalence classes which accomplishes this goal. Another way to say this is that the model corresponds to an *invariant* of the protocol, rather than to a *run* of the protocol.

Another aspect of this model which seems counterintuitive is that it only

Principals: $\mathbb{P} = \{I, A\}$

Data: $\mathbb{D} = \{W, K, U, N, S\}$

Predicates: $\boldsymbol{H} = \{A\}$

$\boldsymbol{K} = \{W, K\}$

Operations: $\bar{I} = W, \quad \bar{A} = K$

$$K_I = W, \quad K_A = K$$

$$W^- = W, \quad K^- = U, \quad U^- = U, \quad N^- = U, \quad S^- = U$$

$$s_{I,I} = K, \quad s_{A,I} = K, \quad s_{I,A} = K, \quad s_{A,A} = U$$

$$(a, b, c) = ((a, b), c)$$

k	W	K	U	N	S
$\{W\}_k$	K	K	U	U	U
$\{K\}_k$	K	K	U	U	U
$\{U\}_k$	U	K	U	U	U
$\{N\}_k$	U	U	U	U	U
$\{S\}_k$	U	U	U	U	U

k	W	K	U	N	S
$[W]_k$	K	K	K	K	K
$[K]_k$	K	K	K	K	K
$[U]_k$	U	U	K	K	K
$[N]_k$	U	U	K	K	K
$[S]_k$	U	U	K	K	K

n	W	K	U	N	S
(W, n)	K	K	N	S	S
(K, n)	K	K	U	S	S
(U, n)	N	N	S	N	N
(N, n)	U	N	U	U	U
(S, n)	N	U	U	U	U

Table 2
A model for the corrected public key Needham-Schroeder protocol

uses a single honest principal. One is tempted to ask "but couldn't there still be a flaw in the protocol when there are two or more honest principals?" Clearly, one could for instance add the existence of at least two honest principals as an additional axiom, and ask whether the intruder could use this additional fact to gain new knowledge. Fortunately this is not the case. Since the protocol, and thus the logic, does not include an equality predicate for principals or for data, we may enlarge the model by creating as many copies of A (and of $\bar{A}$) as we wish, without affecting the validity of formulas in the model. Thus, the protocol is proved secure irrespective of the number of principals involved.

The construction of models such as the one in Table 2 is not a straightforward task. This particular model was constructed by hand and checked by machine. Particularly the definition of the pairing function is quite subtle,

since it must be chosen so as to validate Φ'_i and Φ'_r, without also validating the original Φ_i and Φ_r. In principle, the search for such models could be automated; however, probably some good heuristics are needed to do this successfully. Brute-force methods are likely to fail because of the size of the search space. For instance, even in a 5-element model, there are more than 2^{58} possible binary operations. On the other hand, once a model is constructed, checking its validity is trivial.

3 Using alternate logics

3.1 Classical vs. intuitionistic logic

In the above examples, we have tacitly assumed that we are working in classical first-order logic, and not, for instance, intuitionistic logic. This seems like a conservative assumption, since proofs correspond to potential attacks, and classical logic proves strictly more formulas than intuitionistic logic.

However, it turns out that, for the purpose of protocol descriptions of the kind we have in mind, it does not actually matter whether one works in classical or intuitionistic logic. As we will prove in this section, classical logic and intuitionistic logic coincide when restricted to the kinds of formulas that arise from the formalization of many protocols. To make this precise, consider the following three classes of first-order formulas. Here, by convention, the logical constants ("true" and "false") are counted as atomic formulas.

$$\begin{aligned}
&\text{Type 0 formulas } \alpha ::= \text{atomic formulas} \mid \alpha \wedge \alpha \mid \alpha \vee \alpha \mid \exists x.\alpha \\
&\text{Type 1 formulas } \beta ::= \alpha \mid \beta \wedge \beta \mid \beta \vee \beta \mid \exists x.\beta \mid \forall x.\beta \mid \alpha \to \beta \\
&\text{Type 2 formulas } \gamma ::= \alpha \mid \gamma \wedge \gamma \mid \forall x.\gamma \mid \beta \to \gamma
\end{aligned}$$

Theorem 3.1 *Any Type 2 formula is provable in classical logic if and only if it is provable in intuitionistic logic.*

Proof. The right-to-left implication is trivial. The other implication is easiest seen in terms of Kripke models. In a given Kripke model, we write $s \vdash \phi$ if the formula ϕ is satisfied at world s in the usual Kripke sense. We write $s \vdash_{\text{loc}} \phi$ if ϕ is satisfied at world s locally, i.e., when the world s (with its local assignment of truth values to atomic formulas) is considered as a classical first-order model. The following properties are straightforward to prove by induction on formulas.

(i) For any Type 0 formula α, and any s, $s \vdash \alpha \iff s \vdash_{\text{loc}} \alpha$.

(ii) For any Type 1 formula β, and any s, $s \vdash \beta \Rightarrow s \vdash_{\text{loc}} \beta$.

(iii) For any Type 2 formula γ, and any s, $(\forall t \geq s. t \vdash_{\text{loc}} \gamma) \Rightarrow s \vdash \gamma$.

Now suppose γ is a Type 2 formula provable in classical logic. Then by 3., γ is satisfied at any world s in any Kripke model. By completeness of Kripke models, it follows that γ is intuitionistically provable. □

Note that all axioms in Table 1 are of Type 1, as are the protocol-specific axioms. The formula Ψ is of Type 0. Thus, the implication "$\Xi \wedge \Phi_i \wedge \Phi_r \to \Psi$" is of Type 2, and the Theorem applies to it.

Remark 3.2 Any Type 2 formula γ is (intuitionistically as well as classically) equivalent to a conjunction of formulas of the form $\forall \boldsymbol{x}\,(\beta_1, \ldots, \beta_n \to \alpha)$. Thus the statement of the Theorem is equivalent to saying: for a set of Type 1 axioms, the intuitionistic Type 0 consequences coincide with the classical Type 0 consequences.

3.2 Linear Logic

When modeling certain protocols, it might sometimes be desirable to have more control over the use of logical resources than classical or intuitionistic logic provide. The use of linear logic in this context was suggested by [2] and [3]. We give a brief, and admittedly contrived, example which illustrates where classical logic might fall short.

Consider a server S which shares a private key $K_{A,S}$ with some honest principal A. S also keeps some secret s, which it will reveal publicly at A's request. Before revealing s, S will attempt to ensure that the request really came from A, by following this protocol:

$$
\begin{array}{ll}
(1) & A \to S : N_a \\
(2) & S \to A : [N_a, N_s]_{K_{A,S}} \\
(3) & A \to S : [x, N_s]_{K_{A,S}} \\
(4) & S \to A : [x+1, N_s]_{K_{A,S}} \\
(5) & A \to S : [N_a+2, N_s]_{K_{A,S}} \\
(6) & S : \qquad s
\end{array}
$$

That is, A sends a nonce N_a, to which S responds with a challenge $[N_a, N_s]_{K_{A,S}}$. Note that only A can decrypt this message. Steps (3) and (4) have no practical use for this protocol, except to illustrate a point. In step (3), A is given the opportunity to send any message of the form $[x, N_s]_{K_{A,S}}$ to S, and S will respond with $[x+1, N_s]_{K_{A,S}}$. Finally, in step (5), A proves her identity by replying to the challenge from step (2), and then S reveals the secret.

It is clear that if the intruder can trick S into going through steps (3) and (4) of the protocol twice, for the *same* nonce N_s, then he could easily fool S into revealing the secret. On the other hand, if S carefully keeps track of which step of the protocol have already been performed, and does not allow steps to be repeated within a single session, then this particular attack is spoiled.

The encoding into classical logic does not allow us to draw this particular distinction on the behavior of S. The above protocol is formalized as follows:

$$
\begin{aligned}
\Phi_s \ : \ & \forall n\,(\boldsymbol{K}\,n \to \exists m\,(\boldsymbol{K}[n,m]_{K_{A,S}} \wedge \forall x\,(\boldsymbol{K}[x,m]_{K_{A,S}} \to \\
& (\boldsymbol{K}[F(x),m]_{K_{A,S}} \wedge (\boldsymbol{K}[F(F(n)),m]_{K_{A,S}} \to \boldsymbol{K}\,s))))).
\end{aligned}
$$

However, in classical logic, we cannot state that the intruder should not use the subformula $\forall x\,(\ldots)$ more than once. Thus we cannot model the situation where S disallows repeating steps of the protocol.

One the other hand, linear logic is able to express this distinction. If we write the formula Φ_s in linear logic, using linear implication and multiplicative conjunction, we can explicitly insert the modality "!" in places where we want to allow replication of subformulas. The following formula may serve to embed the above protocol in affine linear logic:

$$
\begin{aligned}
\Phi'_s \ : \ & !\forall n\,(\boldsymbol{K}\,n \multimap \exists m\,(!\,\boldsymbol{K}[n,m]_{K_{A,S}} \otimes \forall x\,(\boldsymbol{K}[x,m]_{K_{A,S}} \multimap \\
& (!\,\boldsymbol{K}[F(x),m]_{K_{A,S}} \otimes (\boldsymbol{K}[F(F(n)),m]_{K_{A,S}} \multimap !\,\boldsymbol{K}\,s))))).
\end{aligned}
$$

Note that we have inserted "!" in front of the entire axiom, since there might be any number of sessions of a protocol, and in front of positive atomic formulas $\boldsymbol{K}\,m$, since knowledge is permanent even in a linear world. This particular encoding reflects the situation where steps (3) and (4) of the protocol may not be repeated. The other situation, where they may be repeated, can be encoded by inserting an "!" just before $\forall x$.

It should be noted that for the purpose of proving simple correctness properties, such as the secrecy properties considered in this paper, using classical logic is adequate in most cases. Since classical logic is a superset of both intuitionistic and linear logic, its use does not lead to false proofs of correctness; in the worst case, it might fail to establish the correctness of some good protocol.

On the other hand, linear logic has the potential to express correctness requirements beyond the ones considered here. For instance, through the use of "additive" connectives, one can formalize the idea that the intruder may take one action or another, but not both. An application of these ideas to contract signing protocols appears in current work by Chadha and Scedrov.

4 Relationship to other models

In this section, we sketch the relationship of our formalism with similar formalisms that were developed by Compton and Dexter [3] and by Cervesato et al. [2]. Both these earlier works are primarily concerned with proof search, and thus use Horn or Horn-like clauses to encode protocol axioms.

4.1 Relationship to Compton and Dexter

Compton and Dexter's framework is based on Horn clauses [3]. The difference between their system and ours lies in a different nesting of quantifiers. To convert a formula from our formalism to Compton and Dexter's, one first pulls all existential quantifiers to the top level, past any universal quantifiers (this is not a logically equivalent transformation). For instance, the formula Φ_r from Section 2.2 is converted to

$$\exists m\,\forall A,B\,\forall n\,(\boldsymbol{K}\{\bar{A},n\}_{K_B}\to(\boldsymbol{K}\{n,m\}_{K_A}\wedge(\boldsymbol{K}\{m\}_{K_B}\to(\boldsymbol{K}[s_{A,B}]_n\wedge\boldsymbol{K}[s_{A,B}]_m)))).$$

The existentially quantified variable m is then turned into a constant M by skolemization, and the resulting formula is converted into three equivalent Horn clauses:

$$\begin{aligned}
&\forall A,B\,\forall n\ \boldsymbol{K}\{\bar{A},n\}_{K_B} && \to \boldsymbol{K}\{n,M\}_{K_A}\\
&\forall A,B\,\forall n\ \boldsymbol{K}\{\bar{A},n\}_{K_B}\wedge\boldsymbol{K}\{M\}_{K_B} && \to \boldsymbol{K}[s_{A,B}]_n\\
&\forall A,B\,\forall n\ \boldsymbol{K}\{\bar{A},n\}_{K_B}\wedge\boldsymbol{K}\{M\}_{K_B} && \to \boldsymbol{K}[s_{A,B}]_M.
\end{aligned}$$

Finally, the proof search algorithm, which is not very well documented in logical terms, subtly makes use of the fact that M is "session-dependent", and thus may actually depend on A, B, and n. Our analysis suggests that one could achieve the same effect, in a more semantically transparent fashion, by skolemizing the existential quantifier without first pulling it to the top level, and thus replacing m by a function symbol $M(A,B,n)$.

4.2 Relationship to Cervesato et al.

Cervesato et al. give an encoding of protocols in linear logic, using Horn-like formulas (essentially multiple-conclusioned Horn clauses where the right-hand-side may be existentially quantified) [2]. The key step for translating our notation to that of Cervesato et al. is to give explicit names to subformulas. For instance, consider again the formula Φ_r from above. By giving names such as $\boldsymbol{B}_0()$, $\boldsymbol{B}_1(A,B,n,m)$, etc., to some of its subformulas, we can write $\Phi_r=\boldsymbol{B}_0$, where

$$\begin{aligned}
\boldsymbol{B}_0() &= \forall A,B\,\forall n(\boldsymbol{K}\{\bar{A},n\}_{K_B}\to\exists m\,(\boldsymbol{K}\{n,m\}_{K_A}\wedge\boldsymbol{B}_1(A,B,n,m)))\\
\boldsymbol{B}_1(A,B,n,m) &= \boldsymbol{K}\{m\}_{K_B}\to\boldsymbol{B}_2(A,B,n,m)\\
\boldsymbol{B}_2(A,B,n,m) &= \boldsymbol{K}[s_{A,B}]_n\wedge\boldsymbol{K}[s_{A,B}]_m.
\end{aligned}$$

Directing these equalities as left-to-right implications, and then pulling universal quantifiers to the top-level (where we omit them), we obtain the following rewrite rules in the style of Cervesato et al.:

$$\begin{aligned}
&\boldsymbol{B}_0(),\boldsymbol{K}\{\bar{A},n\}_{K_B} && \to \exists m.\ \boldsymbol{K}\{n,m\}_{K_A},\boldsymbol{B}_1(A,B,n,m)\\
&\boldsymbol{B}_1(A,B,n,m),\boldsymbol{K}\{m\}_{K_B} && \to \boldsymbol{B}_2(A,B,n,m)\\
&\boldsymbol{B}_2(A,B,n,m) && \to \boldsymbol{K}[s_{A,B}]_n,\boldsymbol{K}[s_{A,B}]_m.
\end{aligned}$$

This transformation captures the basic idea of Cervesato et al.'s encoding. However, their actual encoding still differs in many details which we do not consider here. For instance, they omit the variables A,B, replace constants such as $K_{A,S}$ by declarations made in an "initialization phase", and they split the predicate $\boldsymbol{K}$ into several predicates N,D,M,C, for network, decomposition, memory, and composition, respectively. However, it appears that these details were created primarily in order to facilitate efficient proof search, and they are probably of lesser significance from a logical point of view.

5 Conclusions and future work

We have presented a simple, adversary-centric framework for modeling cryptographic protocols in first-order logic. We have demonstrated the feasibility of the approach by applying it to one of the simplest known protocols with an interesting flaw, the public-key Needham Schroeder protocol. The author admits that this particular protocol has already been over-analyzed in the literature; however, it still serves as a useful illustration of the concepts discussed in this paper.

One new contribution of this paper is the use of first-order models to prove the correctness of protocols. The apparent simplicity of this method, and particularly the fact that there is a single "witness" for the correctness of the protocol, is in contrast to some other methods that have been used to establish similar results, including proof search and model checking techniques. However, the simplicity of the specific countermodel used also causes an uneasy feeling. It highlights the strong assumptions that were used in the formalization of the correctness property of the protocol, particularly the black-box assumption.

The adversary-centric framework seems particularly suited for formalizing secrecy properties. Authentication properties can also be modeled, at least to the extent that they can be reduced to secrecy properties. It is an interesting question whether the approach can be extended to other correctness requirements for protocols, such as agreement requirements used in e-commerce protocols, non-repudiation requirements, etc.

Another interesting question is whether one can formalize more sophisticated protocol features, such as time stamps, and protocol requirements, such as robustness under the disclosure of session-specific data. Our framework can be modified to include a sort of "session identifiers", and a predicate which declares certain sessions to be "corrupted". The axioms can be modified so that corrupted session keys are immediately disclosed. A useful security property, which can be formulated in this framework, is that the intruder should not be able to learn secrets that are associated with a non-corrupted session, even if other sessions of the same protocol have been corrupted.

References

[1] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.

[2] I. Cervesato, N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In R. Gorrieri, editor, *Proceedings of the 12th IEEE Computer Security Foundations Workshop, CSFW '99*. IEEE Computer Society Press, 1999.

[3] K. J. Compton and S. Dexter. Proof techniques for cryptographic protocols. In *Proceedings of ICALP '99*, pages 25–39, 1999.

[4] D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 2(29), 1983.

[5] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using CSP and FDR. In *2nd International Workshop on Tools and Algorithms for the Construction and Analysis of Systems*. Springer Verlag, 1996.