Journal of Logical and Algebraic Methods in Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp

A survey of symbolic methods for establishing equivalence-based properties in cryptographic protocols

Stéphanie Delaune [a,*], Lucca Hirschi [b]

[a] CNRS/IRISA, Rennes, France
[b] LSV, CNRS & ENS Cachan, France

ARTICLE INFO

Article history:
Received 29 December 2015
Received in revised form 3 October 2016
Accepted 24 October 2016
Available online xxxx

Keywords:
Cryptographic protocols
Symbolic models
Privacy-related properties
Behavioural equivalence

ABSTRACT

Cryptographic protocols aim at securing communications over insecure networks such as the Internet, where dishonest users may listen to communications and interfere with them. A secure communication has a different meaning depending on the underlying application. It ranges from the confidentiality of a data to *e.g.* verifiability in electronic voting systems. Another example of a security notion is *privacy*.

Formal symbolic models have proved their usefulness for analysing the security of protocols. Until quite recently, most results focused on trace properties like confidentiality or authentication. There are however several security properties, which cannot be defined (or cannot be naturally defined) as trace properties and require a notion of behavioural equivalence. Typical examples are anonymity, and privacy related properties. During the last decade, several results and verification tools have been developed to analyse equivalence-based security properties.

We propose here a synthesis of decidability and undecidability results for equivalence-based security properties. Moreover, we give an overview of existing verification tools that may be used to verify equivalence-based security properties.

1. Introduction

Security protocols are widely used to secure transmissions in various types of networks (*e.g.,* web, wireless devices, *etc.*). They are (often small) concurrent programs relying on cryptographic primitives. The security properties they should achieve are multiple and depend on the context in which they are used. The main problem they have to cope with is to protect communication that are done through insecure, public, channels like the Internet, where dishonest users may listen to communications and interfere with them. This explains why they are notoriously difficult to design and hard to analyse by hand. Actually, many protocols have been shown to be flawed several years after their publication (and deployment). Given the very sensitive contexts in which they are used, establishing the security of these protocols is a very relevant research goal with important economic and societal consequences.

Two main distinct approaches have emerged, starting with the early 1980s attempt of [1], to ground security analysis of protocols on firm, rigorous mathematical foundations. These two approaches are known as the *computational approach* and the *symbolic approach*.

* Corresponding author.
E-mail address: stephanie.delaune@irisa.fr (S. Delaune).

http://dx.doi.org/10.1016/j.jlamp.2016.10.005
2352-2208/

The computational approach models messages as bit-strings; agents and the attacker as probabilistic polynomial time machines; whereas security properties are defined using games played by the attacker who has to be able to distinguish the protocol from an idealised version of it (with a non-negligible probability). It is generally acknowledged that security proofs in this model offer powerful security guarantees. A serious downside of this approach however is that even for small protocols, proofs are usually long, difficult, tedious, and highly error prone. Moreover, due to the high complexity of such a model, automating such proofs is a very complex problem that is still in its infancy (see *e.g.* [2]).

By contrast, the symbolic approach, which is the one targeted by this survey, makes strong assumptions on cryptographic primitives (*i.e.,* black-boxed cryptography assumption) but fully models agents' interactions and algebraic properties of these primitives. For instance, symmetric encryption and decryption are modelled as function symbols enc and dec along with the equations $\mathsf{dec}(\mathsf{enc}(m,k),k) = m$. This means that, without the corresponding key k, it is simply impossible to get back the plaintext m from the cipher-text $\mathsf{enc}(m,k)$. This does not mean however that protocols relying on these primitives are necessarily secure. There can still remain some *logical attacks* like *e.g.* a man-in-the-middle attack or a reflection attack. Although less precise, this symbolic approach benefits from automation and can thus target more complex protocols than those analysed using the computational approach. Moreover, a line of work known as *computational soundness* aims at spanning the gap between these two approaches by establishing that, in some cases, security guarantees in the symbolic model imply security guarantees in the computational one. This line has been initiated by Abadi and Rogaway [3] and has received much attention since then (see [4] for a survey on computational soundness).

Until the early 2000s, most works from the symbolic approach were focusing on *trace properties*, that is, statements that something bad never occurs on any execution trace of a protocol. Secrecy and authentication are typical examples of trace properties: a data remains confidential if, for any execution, the attacker is not able to produce the data from its observations. But many other properties like strong secrecy, unlinkability or anonymity are not defined as trace properties. These properties are usually defined as the fact that an observer cannot distinguish between two situations, and require a notion of *behavioural equivalence*. Roughly, two protocols are equivalent if an attacker cannot observe whether he is interacting with one or the other. In this survey, we shall focus on equivalence-based security properties.

There exist other approaches out of the scope of this survey that do not strictly follow the symbolic approach nor the computational one but are able to verify notions of behavioural equivalence. A recent approach proposes to define a computationally complete symbolic attacker by axiomatizing what the attacker can *not* do [5]. This approach has been recently extended to deal with a notion of behavioural equivalence [6]. Another work proposed semi-automatic proof of vote privacy using type-based verification [7]. This has been done using the tool Rf*, where protocols are modelled using code-based cryptographic abstractions and security properties are encoded as *refinement types* [8]. Security is achieved by type checking the protocol.

Outline. In Section 2, we give an informal presentation of different cryptographic primitives after which we describe the Basic Access Control (BAC) protocol from the e-passport application, and some of its logical attacks. Section 3 describes various security properties that one may want to verify. In Section 4, we give a formal model, following the symbolic approach, for messages, protocols and equivalence properties. Section 5 is dedicated to the existing methods and tools for verifying equivalence-based properties. We conclude in Section 6.

2. What is a cryptographic protocol?

A cryptographic protocol can be seen as a list of rules that describe executions; these rules specify the emissions and receptions of messages by the actors of the protocols called *agents*. These protocols use as basic building blocks cryptographic primitives such as symmetric/asymmetric encryptions, signatures, and hash functions. For a long time, it was believed that designing a strong encryption scheme was sufficient to ensure secure message exchanges. Starting from the 1980s, researchers understood that even with perfect encryption schemes, message exchanges were still not necessarily secure. This fact will be illustrated in Section 2.2, but we first briefly explain the most standard cryptographic primitives together with their fundamental properties.

2.1. Cryptographic primitives

Cryptographic primitives provide fundamental properties and are used to develop more complex tools called cryptographic protocols, which guarantee one or more high-level security properties.

Symmetric encryption. Symmetric cryptography refers to encryption methods in which both the sender and the receiver share the same key. For instance, the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) are symmetric encryption schemes which have been designated cryptography standards by the US government in 1976 and 2002 respectively.

A significant disadvantage of symmetric ciphers is the key management necessary to use them securely. Each distinct pair of communicating parties must share a different key. Therefore, the number of required keys increases as the square of the number of network members, which very quickly requires complex key management schemes to keep them all straight and secret. The difficulty of securely establishing a secret key between two communicating parties, when a secure channel

Please cite this article in press as: S. Delaune, L. Hirschi, A survey of symbolic methods for establishing equivalence-based properties in cryptographic protocols, J. Log. Algebraic Methods Program. (2016), http://dx.doi.org/10.1016/j.jlamp.2016.10.005