Verifying Secrecy by Abstract Interpretation

L. Bozga & Y. Lakhnech & M. Périn

Vérimag
2, avenue de Vignate
38610 Gires, France
{lbozga,lakhnech,perin}@imag.fr

1. Introduction

At the heart of almost every computer security architecture is a set of cryptographic protocols that use cryptography to encrypt and sign data. They are used to exchange confidential data such as pin numbers and passwords, to authentify users or to guarantee anonymity of participants. It is well known that even under the idealized assumption of perfect cryptography, logical flaws in the protocol design may lead to incorrect behavior with undesired consequences. Maybe the most prominent example showing that cryptographic protocols are notoriously difficult to design and test is the Needham-Schroeder protocol for authentification. It has been introduced in 1978 [19]. An attack on this protocol has been found by G. Lowe using the CSP model-checker FDR in 1995 [13]; and this led to a corrected version of the protocol [14]. Consequently there has been a growing interest in developing and applying formal methods for validating cryptographic protocols [15, 7]. Most of this work adopts the so-called Dolev and Yao model of intruders. This model assumes idealized cryptographic primitives and a nondeterministic intruder that has total control of the communication network and capacity to forge new messages. It is known that reachability is undecidable for cryptographic protocols in the general case [10], even when a bound is put on the size of messages [9]. Because of these negative results, from the point of view of verification, the best we can hope for is either to identify decidable sub-classes as in [1, 21, 16] or to develop correct but incomplete verification algorithms as in [18, 12, 11].

In this talk, we present a correct but, in general, incomplete verification algorithm to prove secrecy without putting any assumption on messages nor on the number of sessions. Proving secrecy means proving that secrets, which are pre-defined messages, are not revealed to unauthorized agents. Our contribution is two fold:

1. We define a concrete operational model for cryptographic protocols, that are given by a set of roles with associated transitions. In general, each transition consists in reading a message from the network and sending a message. Our semantic model allows an unbounded number of sessions and participants, where a participant can play different roles in parallel sessions. Secrets are then specified by messages and are associated to sessions. All of this makes our model infinite. Therefore, we introduce a general and generic abstraction that reduces the problem of secrecy verification for all possible sessions to verifying a secret in a model given by a set of constraints on the messages initially known by the intruder and a set of rules that describe how this knowledge evolves. Roughly speaking, the main idea behind this abstraction step is to fix an arbitrary session running between arbitrary agents. Then, to identify all the other agents as well as their cryptographic keys and nonces. Thus, suppose we are considering a protocol where each session involves two participants playing the role p_1, respectively, p_2. We fix arbitrary participants, A and B, and an arbitrary session i_0. We identify all participants other than A and B and also identify all sessions in which neither A nor B are involved. Concerning the sessions, where A or B are involved except i_0, we make the following identifications:

   - all sessions where A plays the role p_1 (resp. p_2), B plays the role p_2 (resp. p_1),

- all sessions where A (resp. B) plays the role of p_1 (resp. p_2) and the role p_2 (resp. p_1) is played by a participant not in $\{A, B\}$, etc...

Identifying sessions means also identifying the nonces and keys used in these sessions. This gives us a system described as a set of transitions that can be taken in any order and any number of times but which refer to a finite set of atomic messages. We then have to prove that the secret is not revealed by these rules. Here, we should emphasize that, for instance, the methods of [6, 11] can profit from this abstraction as the obtained abstract system can be, in some cases, taken as starting point.

2. Our second contribution is an original method for proving that a secret is not revealed by a set of rules that model how the initial set of messages known by the intruder evolves. In contrast to almost all existing methods, we do not try to compute or approximate the sets of messages that can be known by the intruder. Our algorithm is rather based on the notion of "the secret being *guarded*, or *kept under hat* by a message". For example, suppose that our secret is the nonce N_B and consider the message $\{\{N_A, N_B\}_{k_B}\}_k$. Then, N_B is guarded by $\{N_A, N_B\}_{k_B}$, if the inverse k_B^{-1} of k_B is not known by the intruder. The idea is then to compute a set of guards that will keep the secret unrevealed in all sent messages and such that the inverses of the keys used in this set are also secrets. The difficulty here is that this set is, in general, infinite. Therefore, we introduce *pattern terms* which are terms used to represent sets of guards. For instance the pattern term $\{X, X_S\}_{k_B}$ says that the secret will be guarded in any message $\{m, m'\}_{k_B}$, where the secret is not a sub-message of m but may be a sub-message of m'. The problem is, however, that there might be a rule $\{I, Y\}_{k_B} \to Y$ that will send Y unencrypted to the intruder if (s)he produces the message $\{I, Y\}_{k_B}$. Hence, the pattern $\{X, X_S\}_{k_B}$ will guard the secret except when X is I. Using this idea, we develop an algorithm that computes a stable set of pattern terms that guard the secrets in all sent messages. We developed a prototype in Caml that implements this method. We have been able to verify several protocols taken from [4] including, for instance, the corrected version of the Needham-Schroeder protocol, different versions of Yahalom, as well as Otway-Rees.

Related work Dolev, Even and Karp introduced the class of ping-pong protocols and showed its decidability. The restriction put on these protocols are, however, too restrictive and none of the protocols of [4] falls in this class. Recently, Comon, Cortier and Mitchell [6] extended this class allowing pairing and binary encryption while the use of nonces still cannot be expressed in their model. Reachability is decidable for the bounded number of sessions [1, 21, 16] or when nonce creation is not allowed and the size of messages is bounded [9]. For the general case, on one hand model-checking tools have been applied to discover flaws in cryptographic protocols [13, 22, 17, 5]. The tool described in [5] is a model-cheker dedicated to cryptograhic protocols. On the other hand, methods based on induction and theorem proving have been developed (e.g. [20, 3, 8]). Closest to our work are partial algorithms based on abstract interpretation and tree automata that have been presented in [18, 12, 11]. The main difference is, however, that we do not compute the set of messages that can be known by the intruder but a set of guards as explained above.

Another remark is that the operational semantics we provide in this paper is well-suited to investigate and relate several existing models that one can find in the literature such as the semantics using clauses to model transitions (e.g. [2]) or linear-logic (e.g. [9]). In an ongoing work, we use this semantics to prove relationships between these models.

References

[1] R. M. Amadio et D. Lugiez. On the reachability problem in cryptographic protocols. In *International Conference on Concurrency Theory*, volume 1877, pages 380–394, 2000.

[2] B. Blanchet. Abstracting cryptographic protocols by Prolog rules (invited talk). In Patrick Cousot, editor, *8th International Static Analysis Symposium (SAS'2001)*, volume 2126 of *Lecture Notes in Computer Science*, pages 433–436, Paris, France, July 2001. Springer Verlag.

[3] D. Bolignano. Integrating proof-based and model-checking techniques for the formal verification of cryptographic protocols. *Lecture Notes in Computer Science*, 1427:77–??, 1998.

[4] J. Clark et J. Jacob. A survey on authentification protocol. `http://www.cs.york.ac.uk/~jac/papers/drareviewps.ps`, 1997.

[5] E.M. Clarke, S. Jha, and W. Marrero. Using state space exploration and a natural deduction style message derivation engine to verify security protocols. In *IFIP Working Conference on Programming Concepts and Methods (PROCOMET)*, 1998.

[6] H. Comon, V. Cortier, and J. Mitchell. Tree automata with one memory, set constraints, and ping-pong protocols. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*. Springer, 2001.

[7] H. Comon et V. Shmatikov. Is it possible to decide whether a cryptographic protocol is secure or not? *Journal of Telecommunications and Information Technology.*, paraître, 2002.

[8] V. Cortier, J. Millen, and H. Rueß. Proving secrecy is easy enough. In *Proc. 14th IEEE Computer Security Foundations Workshop (CSFW'01), Cape Breton, Nova Scotia, Canada, June 2001*, pages 97–110. IEEE Comp. Soc. Press, 2001.

[9] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Undecidability of bounded security protocols. In N. Heintze and Proceedings E. Clarke, editors, editors, *Workshop on Formal Methods and Security Protocols — FMSP, Trento, Italy, July 1999.*, 1999.

[10] S. Even et O. Goldreich. On the security of multi-party ping pong protocols. Technical report, Israel Institute of Technology, 1983.

[11] T. Genet et F. Klay. Rewriting for Cryptographic Protocol Verification. In *Proceedings 17th International Conference on Automated Deduction*, volume 1831 of *Lecture Notes in Artificial Intelligence*. Springer-Verlag, 2000.

[12] Jean Goubault-Larrecq. A method for automatic cryptographic protocol verification. In Dominique Méry, Beverly Sanders, editors, *Fifth International Workshop on Formal Methods for Parallel Programming: Theory and Applications (FMPPTA 2000)*, number 1800 in Lecture Notes in Computer Science. Springer-Verlag, 2000.

[13] G. Lowe. An attack on the Needham-Schroeder public-key authentification protocol. *Information Processing Letters*, 56(3):131–133, November 1995.

[14] G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using fdr. In *TACAS*, number 1055 in Lecture Notes in Computer Science, pages 147–166, 1996.

[15] C. Meadows. Invariant generation techniques in cryptographic protocol analysis. In *PCSFW: Proceedings of The 13th Computer Security Foundations Workshop*. IEEE Computer Society Press, 2000.

[16] J. Millen et V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *Proc. 8th ACM Conference on Computer and Communications Security (CCS '01)*, pages 166–175, 2001.

[17] J. C. Mitchell, M. Mitchell, et U. Stern. Automated analysis of cryptographic protocols using mur. In *Proceedings of the 1997 Conference on Security and Privacy (S&P-97)*, pages 141–153, Los Alamitos, May 4–7 1997. IEEE Press.

[18] D. Monniaux. Decision procedures for the analysis of cryptographic protocols by logics of belief. In *12th Computer Security Foundations Workshop*. IEEE, 1999.

[19] R.M. Needham and M.D. Schroeder. Using encryption for authentication in large networks of computers. *CACM*, 21(12):993–999, 1978.

[20] L. Paulson. Proving properties of security protocols by induction. In *10th IEEE Computer Security Foundations Workshop (CSFW '97)*, pages 70–83, June 1997. IEEE.

[21] M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is np-complete. In *14th IEEE Computer Security Foundations Workshop (2001), pp. 174–190.*, 2001.

[22] S. Schneider. Verifying authentication protocols with CSP. In *10th IEEE Computer Security Foundations Workshop (CSFW '97)*, pages 3–17, June 1997. IEEE.