Techniques for verification of an unbounded number of sessions

David Baelde[1], Stéphanie Delaune[2], and Steve Kremer[3]

[1] LSV, CNRS & ENS Cachan, Université Paris Saclay
[2] IRISA & CNRS
[3] LORIA, Inria Nancy & CNRS & Université de Lorraine

Abstract. This deliverable concerns the Task 2 of the Sequoia project: *Automated verification of process equivalences.* We have proposed new techniques and tools to verify equivalence based properties. Relying on these results, we have analysed several protocols among them some protocols from the e-passport application, as well as several e-voting protocols.

The results presented in this report are based on results that have been published in [5, 7, 1]– works that have been supported by the Sequoia project.

1 Introduction

Security protocols are used in many of our daily-life applications, and our privacy largely depends on their design. Since security protocols are notoriously difficult to design and analyze, formal verification techniques are important. These techniques have become mature and have been used with success to analyse several protocols. For example, a flaw has been discovered in the Single-Sign-On protocol used *e.g.* by Google Apps. It has been shown that a malicious application could very easily get access to any other application (*e.g.* Gmail or Google Calendar) of their users [2]. This flaw has been found when analysing the protocol using formal methods, abstracting messages by a term algebra and using the Avantssar validation platform [3].

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. Here, we focus on equivalence-based security properties.

Security protocols used in practice are more and more complex and, often, formal analyzers are not powerful enough to handle all kinds of attacks and

features of protocols. For instance, ProVerif [4] is able to deal with an infinite number of sessions but can only check a strong form of equivalence, namely *diff-equivalence*. APTE and AKISS are devoted to the problem of checking trace equivalence but they are limited to a finite number of sessions. Another line of research has focused on the design of type systems for cryptographic protocol analysis. Type systems proved capable to enforce even observational equivalence relations, such as secrecy. Although they look promising, type systems have not previously been used in the context of e-voting protocols. This task is challenging since, for guiding the type-checking procedure, one needs to develop a dedicated logical theory, capturing the structure of e-voting systems and the associated security and privacy properties.

In this report, we present two main results. We first explain how we have leveraged the existing automatic verification tool ProVerif to go beyond diff-equivalence (see Section 2). This technique has been implemented in a tool UKANO and allows one to analyse unlinkability and anonymity for a large class of 2-party protocols. In Section 3, we explain the techniques we have developped to analyse e-voting protocols. These protocols aim at achieving a wide range of sophisticated security properties and, consequently, commonly employed advanced cryptographic primitives. This makes their design as well as rigorous analysis quite challenging, thus requires developping dedicated techniques.

2 A method to check unlinkability and anonymity

We identify a large class of 2-party protocols (simple else branches, arbitrary cryptographic primitives) and we devise two conditions that imply unlinkability and anonymity for an unbounded number of sessions. We show how these two conditions can be automatically checked using the ProVerif tool, and we provide tool support for that, namely Ukano. We have analyzed several protocols, among them the Basic Access Control (BAC) protocol as well as the Password Authenticated Connection Establishment (PACE) protocol that are both used in e-passports. We notably establish the first proof of unlinkability for the BAC protocol followed by the Passive Authentication (PA) and Active Authentication (AA) protocols. We also report on an attack that we found on the PACE protocol, and another one that we found on the LAK protocol whereas it is claimed untraceable in [8]. Our conditions appear to be rather tight: we provide an attack every time one of them is not satisfied.

We now give an intuitive overview of these two conditions. In order to do this, assume that we want to design a mutual authentication protocol between a tag T and a reader R based on symmetric encryption, and we want this protocol to be unlinkable. We note $\{m\}_k$ the symmetric encryption of a message m with a key k and we assume that k is a symmetric key shared between T and R.

A first attempt to design such a protocol is presented using so-called Alice & Bob notation as follows (n_R is a fresh nonce):

$$1.\ R \to T : n_R$$
$$2.\ T \to R : \{n_R\}_k$$

This first attempt based on a challenge-response scheme is actually linkable. Indeed, an active attacker who systematically intercepts the nonce n_R and replaces it by a constant will be able to infer whether the same tag has been used in different sessions or not by comparing the answers he receives. Here, the tag is linkable because, for a certain behavior (possibly malicious) of the attacker, some relations between messages leak information about the agents that are involved in the execution. Our first condition, namely *frame opacity*, actually checks that all outputted messages have only trivial relations that can therefore not be exploited by the attacker.

Our second attempt takes the previous attack into account and randomizes the tag's response and should achieve mutual authentication by requiring that the reader must answer to the challenge n_T. This protocol can be as follows:

$$\begin{array}{l} 1.\ R \to T : n_R \\ 2.\ T \to R : \{n_R, n_T\}_k \\ 3.\ R \to T : \{n_T\}_k \end{array}$$

Here, Alice & Bob notation shows its limit. It does not specify how the reader and the tag are supposed to check that the messages they received are of the expected form, and how they should react when the messages are not well formed. This has to be precisely defined, since unlinkability depends on it. For instance, assume the tag does not check that the message he receives at step 3 contains n_T, and aborts the session if the received message in not encrypted with its own k. In such an implementation, an active attacker can eavesdrop a message $\{n_T\}_k$ sent by R to a tag T, and try to inject this message at the third step of another session played by T'. The tag T' will react by either aborting or by continuing the execution of this protocol. Depending on the reaction of the tag, the attacker will be able to infer if T and T' are the same tag or not.

In this example, the attacker adopts a malicious behavior that is not detected immediately by the tag who keeps executing the protocol. The fact that the tag passes successfully a conditional reveals crucial information about the agents that are involved in the execution. Our second condition, namely *well-authentication*, basically requires that when an execution deviates from the honest one, the agents that are involved cannot successfully pass a conditional.

Our main theorem states that these two conditions, frame opacity and well-authentication, are actually sufficient to ensure both unlinkability and anonymity. This theorem is of interest as our two conditions are fundamentally simpler than the targeted properties: frame opacity can be expressed using diff-equivalence and well-authentication is a trace property. In fact, they are both in the scope of existing automatic verification tools like ProVerif.

This result has been obtained by Lucca Hirschi, David Baelde and Stéphanie Delaune and has been published in the proceedings of S&P'16 [7]. An extended version is currently under preparation.

3 The special case of electronic voting protocols

E-voting protocols aim at achieving a wide range of sophisticated security properties and, consequently, commonly employ advanced cryptographic primitives. This makes their design as well as rigorous analysis quite challenging. As a matter of fact, existing automated analysis techniques, which are mostly based on automated theorem provers, are inadequate to deal with commonly used cryptographic primitives, such as homomorphic encryption and mix-nets, as well as some fundamental security properties, such as verifiability.

We propose a novel approach based on refinement type systems for the automated analysis of e-voting protocols. Specifically, we design a generically applicable logical theory which, based on pre- and post-conditions for security-critical code, captures and guides the type-checker towards the verification of two fundamental properties of e-voting protocols, namely, vote privacy and verifiability. We further develop a code-based cryptographic abstraction of the cryptographic primitives commonly used in e-voting protocols, showing how to make the underlying algebraic properties accessible to automated verification through logical refinements. Finally, we demonstrate the effectiveness of our approach by developing the first automated analysis of Helios, a popular web-based e-voting protocol, using an off-the-shelf type-checker.

To ease the analysis of e-voting protocols, we also propose a reduction result that avoids the need to consider an arbitrary number of voters. More precisely, we identify a class of voting protocols for which only a small number of agents needs to be considered: if there is an attack on vote privacy then there is also an attack that involves at most 3 voters (2 honest voters and 1 dishonest voter). In the case where the protocol allows a voter to cast several votes and counts, e.g., only the last one, we also reduce the number of ballots required for an attack to 10, and under some additional hypotheses, 7 ballots. Our results are formalised and proven in a symbolic model based on the applied pi calculus. We illustrate the applicability of our results on several case studies, including different versions of Helios and Prêt-à-Voter, as well as the JCJ protocol. For some of these protocols we can use the ProVerif tool to provide the first formal proofs of privacy for an unbounded number of voters.

This first result has been obtained by Véronique Cortier, Fabienne Eigner, Steve Kremer, Matteo Maffei, and Cyrille Wiedling, and has been published at POST'15[5]. The second one has been obtained by Myrto Arapinis, Véronique Cortier, and Steve Kremer. It has been published at ESORICS'16[1].

4 Conclusion

We have proposed several results that allows one to establish equivalence based security properties. These techniques allow one to analyse a wide range of security protocols, but some protocols are still out of range of our techniques. For instance, the e-voting protocol recently deployed in Norway for political elections

has been analysed in [6] but the proof was completely done by hand. Indeed, due to the algrabraic properties of the encryption scheme, none of the techniques proposed so far is suitable to analyse this protocol automatically.

References

1. M. Arapinis, V. Cortier, and S. Kremer. When are three voters enough for privacy properties? In I. Askoxylakis, S. Ioannidis, S. Katsikas, and C. Meadows, editors, *Proceedings of the 21st European Symposium on Research in Computer Security (ESORICS'16)*, volume 9879 of *Lecture Notes in Computer Science*, pages 241–260, Heraklion, Crete, Sept. 2016. Springer.
2. A. Armando, R. Carbone, L. Compagna, J. Cuéllar, and M. L. Tobarra. Formal analysis of SAML 2.0 web browser single sign-on: breaking the SAML-based single sign-on for google apps. In *Proc. 6th ACM Workshop on Formal Methods in Security Engineering (FMSE 2008)*, pages 1–10. ACM Press, 2008.
3. A. Armando et al. The AVANTSSAR platform for the automated validation of trust and security of service-oriented architectures. In *Proc. 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'12)*, volume 7214, pages 267–282. Springer, 2012.
4. B. Blanchet and A. Podelski. Verification of cryptographic protocols: Tagging enforces termination. In *Foundations of Software Science and Computation Structures (FoSSaCS'03)*.
5. V. Cortier, F. Eigner, S. Kremer, M. Maffei, and C. Wiedling. Type-based verification of electronic voting protocols. In *Proceedings of the 4th Conference on Principles of Security and Trust (POST'15)*, volume 9036 of *Lecture Notes in Computer Science*, pages 303–323, London, UK, Apr. 2015. Springer.
6. V. Cortier and C. Wiedling. A formal analysis of the norwegian e-voting protocol. *Journal of Computer Security*, 25(15777):21–57, 2017.
7. L. Hirschi, D. Baelde, and S. Delaune. A method for verifying privacy-type properties: the unbounded case. In M. Locasto, V. Shmatikov, and Ú. Erlingsson, editors, *Proceedings of the 37th IEEE Symposium on Security and Privacy (S&P'16)*, San Jose, California, USA, May 2016. IEEE Computer Society Press.
8. T. van Deursen and S. Radomirovic. Attacks on rfid protocols. *IACR Cryptology ePrint Archive*, 2008:310, 2008.