Decision procedures for equivalence based properties

Stéphanie Delaune and Steve Kremer

[1] LSV, ENS Cachan & CNRS & Inria Saclay Île-de-France
[2] LORIA, Inria Nancy Grand Est

The results presented in this report are based on results that have been published in [13, 9, 10, 14] – works that have been supported by the VIP project and that have been added at the end of this report.

Abstract. This deliverable concerns the Task 3 of the VIP project: *Algorithmic and decidability issues.* This is an intermediary report whose aim is to sum up progress that have been made to design algorithms to perform abstract analysis of security protocols against formally stated security properties.

1 Introduction

Security protocols are widely used today to secure transaction that rely on public channels like the Internet. It is therefore essential to obtain as much confidence as possible in their correctness. Starting in the 80s, many works have been devoted to the use of formal methods to analyse the security of these protocols (*e.g.* [15, 18]). In the case of a bounded number of sessions, secrecy preservation is co-NP-complete [18], and for an unbounded number of sessions, several decidable classes have been identified (*e.g.* [15]). Many tools have also been developed to automatically verify cryptographic protocols (*e.g.* AVISPA [4], ProVerif [7]).

Until recently, most efforts and successes only concerned trace properties, *i.e.* security properties that can be checked on each individual sequence of messages corresponding to an execution of the protocol. Secrecy (at least weak forms of secrecy) and authentication are typical examples of trace properties. There are however several security properties, which cannot be defined as trace properties and require a notion of *behavioural equivalence*. We focus here on the notion of *trace equivalence* which is well-suited for the analysis of security protocols. Intuitively, two processes P and Q are trace equivalent, denoted $P \approx_t Q$, if any experiment performed by an attacker on both processes leads to the emission of two sequences of messages that are indistinguishable, *i.e.* the attacker cannot observe any difference between these two sequences. Originally, observational equivalence is a bisimulation-based equivalence notion that has been introduced as a proof technique for trace equivalence [2]. Trace equivalence is probably more adequate to the formalization of privacy-type properties. In this project, we are thus mainly interested in automating the proofs of trace equivalence, but we may also rely on some other notions of equivalence as proof techniques.

State of the art at the beginning of the VIP *project.* A line of works consists in designing stronger notions of equivalences that imply observational equivalence (and thus trace equivalence). This approach has for instance been used in [5, 6, 19, 11], relying on constraint solving techniques. ProVerif implements an algorithm, based on Horn clauses and dedicated resolution strategies, which is able to establish the observational equivalence between two processes written in the applied pi calculus [8]. However, all these methods check a stronger equivalence than observational equivalence and fail on some simple toy examples. Unfortunately, this is exactly the kind of situations we encountered in several case studies, *e.g.* the private authentication protocol [1], and e-passport protocols [3].

Another limitation of the existing approaches is the fact that they are not well-suited to analyse protocols that rely on some non standard cryptographic primitives. For instance, the decision procedures described in [19, 11] only work for signatures, pairs, and encryptions. This is quite restrictive in particular when studying electronic voting protocols that often rely on some non standard primitives such as blind signatures (*e.g.* FOO protocol), or trapdoor bit commitment (*e.g.* Okamoto protocol). This is also a limitation for analysing RFID protocols that often rely on exclusive or. A brief description of these case studies can be found in [16].

Contributions. We propose proof techniques for analysing/deciding equivalence-based properties. Since replication very quickly yields to undecidability even in the simpler case of reachability properties such as secrecy, to get decision procedures, we first focus on finite processes, *i.e.* processes without replication (see Section 2). Then, in Section 3, we briefly describe the results we have obtained for processes with replication.

2 The case of finite processes

We identify several difficulties that we tackle separately. First some case studies require us to consider non trivial else branches whereas some other rely on some non standard cryptographic primitives.

2.1 Dealing with else branches

The applied pi calculus provides an elegant and convenient formalism for expressing security protocols. However, its syntax and semantics (in particular name restriction and parallel composition) do not ease the verification task. In contrast, constraint systems are a much simpler and more pure formalism which captures exactly the core of protocol executions. In particular, constraint systems have shown their usefulness for analysing security protocols, at least for secrecy and authentication properties. Some attempts have also been made to analyse privacy-type properties in this framework. It has been shown (see [5])

that symbolic equivalence between pairs of positive constraint systems is decidable. Even though this result is a bit restrictive, it is sufficient to show decidability of trace equivalence for simple processes without replication and only trivial else branches. Since then, a new procedure more amenable to automation has been developed (see [11]) and implemented. This procedure only works for the standard cryptographic primitives. Our goal is to go beyond this class of simple processes with trivial else branches in order to analyse the protocols described in [16].

To achieve this, in [13], we show a reduction result for general processes without replication and for arbitrary equational theories. We reduce the decidability of trace equivalence (for finite processes) to deciding symbolic equivalence between sets of constraint systems. To transfer executions from the applied pi calculus to constraint systems, we introduce an intermediate calculus. Then, we propose an alogithm to decide equivalence of sets of constraint systems for the fixed theory corresponding to standard cryptographic primitives (symmetric and asymmetric encryption, signatures, hashes). A first version of this work has already been published [12]. We are currently working on an enriched version of this paper and we plan to submit it to a journal.

Prototype. An implementation of this decision procedure has been implemented in a tool called APTE. The tool is implemented in Ocaml (around 12 000 lines). APTE is an open source software and is distributed under GNU General Public Licence 3.0. The tool is available at :

http://projects.lsv.ens-cachan.fr/APTE/.

As expected, APTE checks trace equivalence for processes that use standard primitives (*e.g.* pairing, signatures, hash functions, symmetric and asymmetric encryptions). We can model in particular conditionals (with non-trivial else branches), private channels, and non-deterministic choice. However, we consider processes without replications. In case of failure when establilshing trace equivalence, APTE provides a witness of non-equivalence and gives the actions that have to be done to mount the attack.

The procedure performed well on constraint systems. However, the interleaving step that is required for moving from symbolic equivalence to trace equivalence, is performed in a naive way and it appears that this step is expensive from the computation point of view.

2.2 Dealing with more equational theories

In this direction, our contribution is a new procedure for verifying equivalence properties for processes specified in a cryptographic process calculus (without replication). The messages in the calculus are modeled as terms equipped with an equational theory, similar to the applied pi calculus, but we do not consider else branches.

Our procedure allows to verify trace equivalence on a class of deterministic processes. For non-deterministic processes it may be used to check for two equivalences which over- and under-approximate the standard notion of trace equivalence $\approx_t$ for cryptographic protocols: the under-approximation can be used to prove protocols correct while the over-approximation can be used to rule out incorrect protocols. A novelty of our procedure is that it is based on a fully abstract modeling of symbolic traces for a bounded number of sessions in first-order Horn clauses. This is in contrast to the constraint-solving techniques mentioned above. Our modelling is fully abstract for arbitrary cryptographic primitives that can be modeled as a convergent rewrite system which has the finite variant property. This strictly includes the class of primitives that can be modeled as subterm convergent rewrite systems, and allows us to handle an even larger class of cryptographic primitives. For example, this allows us to handle trapdoor commitment as used by Okamoto for electonic voting. Although we were unable to prove termination of our procedure, we conjecture it to terminate for the class of cryptographic primitives that can be modeled as subterm convergent rewrite systems. Our conjecture is supported by experimental evidence.

Prototype. This decision procedure has been implemented in a tool called AKISS. The tool is implemented in Ocaml (around 3000 lines) and is available at :

https://github.com/ciobaca/akiss.

Relying on this tool, we successfully prove anonymity of the FOO and Okamoto electronic voting protocols. To our knowledge, no other tool can handle these protocols automatically. On a standard modern laptop, AKISS takes less than a minute to carry out the above verification. As in the previous approach, interleaving individual roles of a protocol introduces an exponential blowup on the number of traces to consider. Thus, the tool does not scale vey well.

3 Dealing with processes with replication

To the best of our knowledge, the only verification tool that is able to check equivalence in the active setting for an unbounded number of sessions is the ProVerif tool [8]. Actually, it considers a process equivalence relation that can be checked on a single biprocess. This relation is, however, stronger than process indistinguishability, and hence not suitable in some cases. In particular, we have already observed that the approximations used in ProVerif are not suited for the privacy properties encountered in electronic voting protocols. Some similar problems will occur for other applications such as the private authentication protocol, and the e-passeport protocol. Moreover, ProVerif is not well-suited to decide trace equivalence that seems however to be the right notion in some applications.

3.1 Extension of ProVerif to deal with else branches

Since the notion of equivalence proved by ProVerif is stronger than observational equivalence, it may yield false attacks. Indeed, ProVerif proves equivalences $P \approx Q$ in which P and Q are two variants of the same process obtained by selecting different terms for P and Q. Moreover, ProVerif requires that all tests yield the same result in both processes, in particular the tests of conditional branchings. Thus, for a protocol that does not satisfy this condition, ProVerif will fail to prove equivalence. Unfortunately, many indistinguishable processes do not satisfy this condition. Consider for example the processes:

$$P = \mathsf{in}(c, x).\mathsf{if}\ x = \mathsf{pk}(ska)\ \mathsf{then}\ \mathsf{out}(c, \mathsf{enc}(s, \mathsf{pk}(ska)))\ \mathsf{else}\ \mathsf{out}(c, \mathsf{enc}(n_p, \mathsf{pk}(ska)))$$
$$Q = \mathsf{in}(c, x).\mathsf{if}\ x = \mathsf{pk}(skb)\ \mathsf{then}\ \mathsf{out}(c, \mathsf{enc}(s, \mathsf{pk}(skb)))\ \mathsf{else}\ \mathsf{out}(c, \mathsf{enc}(n_q, \mathsf{pk}(skb)))$$

where all names but c are private and the public keys $\mathsf{pk}(ska)$ and $\mathsf{pk}(skb)$ are public. The protocol P is simply waiting for the public key of the agent A ($\mathsf{pk}(ska)$) on a channel c. If P receives it, then he sends some secret s encrypted with A's public key; otherwise, he sends a fresh nonce n_p encrypted with A's public key on channel c. On the other hand, the protocol Q does similar actions but is waiting for the public key of the agent B ($\mathsf{pk}(skb)$) instead of A. Assuming that the attacker does not have access to the private keys of A and B, then the two protocols are equivalent since the attacker cannot differentiate the different ciphertexts: $\mathsf{enc}(s, \mathsf{pk}(ska))$, $\mathsf{enc}(n_p, \mathsf{pk}(ska))$, $\mathsf{enc}(s, \mathsf{pk}(skb))$, and $\mathsf{enc}(n_q, \mathsf{pk}(skb))$. However, if the intruder sends the public key of the agent A ($\mathsf{pk}(ska)$), then the test of the conditional branching in P will succeed ($\mathsf{pk}(ska) = \mathsf{pk}(ska)$) whereas the test of the same conditional branching in Q will fail ($\mathsf{pk}(ska) \neq \mathsf{pk}(skb)$). Since this test does not yield the same result in both processes, ProVerif will fail to prove the equivalence between P and Q. This false attack also occurs in more realistic case studies, *e.g.*, the private authentication protocol and the Basic Access Control protocol of the e-passport.

Our main contribution consists in addressing the issue of false attacks due to conditional branchings. In particular, we allow function symbols defined by rewrite rules with inequalities as side-conditions, so that we can express tests of conditional branchings directly inside terms. Therefore, we still consider equivalences between processes that differ by the terms they contain, but our term algebra is richer as it can express tests. Hence, we are now able to prove equivalences between processes that take different branches in internal tests, provided that what they do after these tests can be merged into a single process. We show how the original Horn clauses based algorithm of ProVerif can be adapted to our new calculus. Moreover, we provide an automatic procedure that transforms a process into an equivalent process that contains as few conditional branchings as possible, which allows ProVerif to prove equivalence on a larger class of processes.

Prototype. This extension is implemented in ProVerif, which is available at:

http://proverif.inria.fr.

Experimental evaluation showed that the automatic transformation of a biprocess is efficient and returns few biprocesses. In particular, our extension automatically proves anonymity for the private authentication protocol for an unbounded number of sessions. We have also eliminated some false attacks for the Basic Access Control protocol of the e-passport; however, other false attacks remain so we are still unable to conclude for this protocol.

3.2 A first decidability result for processes with replication

We study the decidability of equivalence of security protocols for an unbounded number of sessions. Even in the case of reachability properties such as secrecy, the problem is undecidable in general. We therefore focus on a class of protocols for which secrecy is decidable. This class typically assumes that each protocol rule manipulates at most one variable. Surprisingly, even a fragment of this class (with only symmetric encryption) turns out to be undecidable for equivalence properties. We consequently further assume our protocols to be deterministic (that is, given an input, there is at most one possible output). We show that equivalence is decidable for an unbounded number of sessions and for protocols with standard primitives but pairs. Interestingly, we show that checking for equivalence of protocols actually amounts into checking equality of languages of deterministic pushdown automata. The decidability of equality of languages of deterministic pushdown automata is a difficult problem, shown to be decidable at Icalp'97. We actually characterize equivalence of protocols in terms of equivalence of deterministic generalized real-time pushdown automata, that is deterministic pushdown automata with no epsilon-transition but such that the automata may unstack several symbols at a time. More precisely, we show how to associate to a process P an automata $\mathcal{A}_P$ such that two processes are equivalent if, and only if, their corresponding automata yield the same language and, reciprocally, we show how to associate to an automata $\mathcal{A}$ a process $P_{\mathcal{A}}$ such that two automata yield the same language if, and only if, their corresponding processes are equivalent, that is:

$$P \approx Q \;\Leftrightarrow\; \mathcal{L}(\mathcal{A}_P) = \mathcal{L}(\mathcal{A}_Q), \text{ and } \mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B}) \;\Leftrightarrow\; P_{\mathcal{A}} \approx P_{\mathcal{B}}.$$

Therefore, checking for equivalence of protocols is as difficult as checking equivalence of deterministic generalized real-time pushdown automata.

G. Sénizergues is currently implementing his procedure for pushdown automata. As soon as the tool will be available, we plan to implement our translation, yielding a tool for automatically checking equivalence of security protocols, for an unbounded number of sessions.

4 Conclusion

We now have several procedures to analyse privacy type properties. They all have been implemented (or are currently under implementation). Each procedure has

its own specificities and limitations. In particular, we plan to further study the AKISS procedure to integrate some additional equational theories such as the exclusive or operator. Moreover, we need to cope with the interleaving problem mentioned above. For this, we would like to propose some optimisations to reduce the number of interleavings that have to be considered, and so the number of equivalence between sets of constraint systems that have to be checked. This problem has already been studied in the context of trace properties [17] but seems to be more challenging for trace equivalence. We have already obtained some encouraging results in this direction and we are currently working to integrate this optimisiation into existing verification tools.

References

1. M. Abadi and C. Fournet. Private authentication. *Theoretical Computer Science*, 322(3):427–476, 2004.
2. M. Abadi and A. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Proc. 4th Conference on Computer and Communications Security (CCS'97)*, pages 36–47. ACM Press, 1997.
3. M. Arapinis, T. Chothia, E. Ritter, and M. Ryan. Analysing unlinkability and anonymity using the applied pi calculus. In *Proc. of 23rd IEEE Computer Security Foundations Symposium (CSF'10)*, pages 107–121. IEEE Computer Society Press, 2010.
4. A. Armando et al. The AVISPA Tool for the automated validation of internet security protocols and applications. In *Proc. 17th Int. Conference on Computer Aided Verification (CAV'05)*, volume 3576 of *LNCS*, pages 281–285. Springer, 2005.
5. M. Baudet. Deciding security of protocols against off-line guessing attacks. In *Proc. 12th Conference on Computer and Communications Security (CCS'05)*, pages 16–25. ACM Press, 2005.
6. M. Baudet. *Sécurité des protocoles cryptographiques : aspects logiques et calculatoires*. Phd thesis, École Normale Supérieure de Cachan, France, 2007.
7. B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *Proc. 14th Computer Security Foundations Workshop (CSFW'01)*, pages 82–96. IEEE Comp. Soc. Press, 2001.
8. B. Blanchet, M. Abadi, and C. Fournet. Automated verification of selected equivalences for security protocols. *Journal of Logic and Algebraic Programming*, 75(1):3–51, 2008.
9. R. Chadha, Ş. Ciobâcă, and S. Kremer. Automated verification of equivalence properties of cryptographic protocols. In H. Seidl, editor, *Programming Languages and Systems —Proceedings of the 21th European Symposium on Programming (ESOP'12)*, volume 7211 of *Lecture Notes in Computer Science*, pages 108–127, Tallinn, Estonia, Mar. 2012. Springer.
10. V. Cheval and B. Blanchet. Proving more observational equivalences with proverif. In D. Basin and J. Mitchell, editors, *Proceedings of the 2nd International Conference on Principles of Security and Trust (POST'13*, Lecture Notes in Computer Science, pages 226–246, Roma, Italy, Mar. 2013. Springer.
11. V. Cheval, H. Comon-Lundh, and S. Delaune. Automating security analysis: symbolic equivalence of constraint systems. In *Proc. 5th International Joint Conference on Automated Reasoning (IJCAR'10)*, volume 6173 of *LNAI*, pages 412–426. Springer-Verlag, 2010.

12. V. Cheval, H. Comon-Lundh, and S. Delaune. Trace equivalence decision: Negative tests and non-determinism. In Y. Chen, G. Danezis, and V. Shmatikov, editors, *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*, pages 321–330, Chicago, Illinois, USA, Oct. 2011. ACM Press.
13. V. Cheval, V. Cortier, and S. Delaune. Deciding equivalence-based properties using constraint solving. *Theoretical Computer Science*, 492:1–39, June 2013.
14. R. Chrétien, V. Cortier, and S. Delaune. From security protocols to pushdown automata. In F. V. Fomin, R. Freivalds, M. Kwiatkowska, and D. Peleg, editors, *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP'13) – Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 137–149, Riga, Latvia, July 2013. Springer.
15. D. Dolev and A. C. Yao. On the security of public key protocols. In *Proc. 22nd Symposium on Foundations of Computer Science (FOCS'81)*, pages 350–357. IEEE Computer Society Press, 1981.
16. L. Hirschi and S. Delaune. Description of some case studies. Deliverable VIP 6.1, (ANR-11-JS02-0006), Sept. 2013. 14 pages.
17. S. Mödersheim, L. Viganò, and D. A. Basin. Constraint differentiation: Search-space reduction for the constraint-based analysis of security protocols. *Journal of Computer Security*, 18(4):575–618, 2010.
18. M. Rusinowitch and M. Turuani. Protocol insecurity with finite number of sessions is NP-complete. In *Proc. 14th Computer Security Foundations Workshop (CSFW'01)*, pages 174–190. IEEE Comp. Soc. Press, 2001.
19. A. Tiu and J. E. Dawson. Automating open bisimulation checking for the spi calculus. In *Proc. 23rd IEEE Computer Security Foundations Symposium (CSF'10)*, pages 307–321. IEEE Computer Society Press, 2010.