Decidability of Deterministic Process Equivalence for Finitary Deduction Systems

Benito Fabian Romero Jimenez, Yannick Chevalier

Decidability of Deterministic Process Equivalence for Finitary Deduction Systems

Yannick Chevalier *IRIT,*
Université Toulouse 3
Toulouse, France
yannick.chevalier@irit.fr

Fabian Romero *IRIT,*
Université Toulouse 3
Toulouse, France
benito.romero@irit.fr

***Abstract*—Deciding privacy-type properties of deterministic cryptographic protocols such as anonymity and strong secrecy can be reduced to deciding the symbolic equivalence of processes, where each process is described by a set of possible symbolic traces. This equivalence is parameterized by a deduction system that describes which actions and observations an intruder can perform on a running system.**

We present in this paper a notion of finitary deduction systems. For this class of deduction system, we first reduce the problem of the equivalence of processes with no disequations to the resolution of reachability problem on each symbolic trace of one process, and then testing whether each solution found is solution of a related trace in the other process. We then extend this reduction to the case of generic deterministic finite processes in which symbolic traces may contain disequalities.

***Index Terms*—Cryptographic protocols; formal methods; privacy analysis.**

I. Introduction

Context. Security protocols, *i.e.* protocols in which the messages are cryptographically secured, are a cornerstone of security in distributed applications. The need for optimizing resource utilization and their distributed nature make their design error prone, and formal methods have been applied successfully to detect errors in the past [1]–[4].

Formal models of cryptographic protocols usually present the reader with a dichotomy between the honest agents—translated into a constraint system [5]–[7], clause sets with a subset dedicated to the modeling of honest agents [4], [8] or frames [9], [10]—, and the attacker—modeled by a deduction system expressing its possible actions. In contrast the representation with *symbolic derivation* [11] unifies the honest and dishonest agent models, where agents may perform deductions, nonce creation, tests, and communication actions.

Intuition. First, a trivial remark: since one can construct deduction systems for which reachability is decidable but static equivalence is not [10]. Thus being able to decide reachability does not imply being able to decide static equivalence, which is a special case of process equivalence. However in *e.g.* [3], [6], [12] reachability is reduced to the satisfiability of a constraint system in a bounded number of steps. That problem is decided using constraint transformation rules. A *solved form* is defined as a constraint system in which the attacker just has to instantiate variables by any term he can construct. In practice, the proof of completeness of the procedure consists in assuming the existence of a sequence of deduction steps that satisfies the constraint system, and in proving that as long as one such sequence exists, either the constraint system is in solved form or there exists a transformation rule applicable on the constraint system. Then, one proves that there is no infinite sequence of transformations and that each constraint system not in solved form has only a finite number of possible successors. König's lemma then implies termination of the procedure. The original constraint system is unsatisfiable if and only if the set of leaves in solved form is empty.

Such procedures do much more than simply deciding reachability, as they end with a (possibly empty) set of constraint systems in solved form that, as long as the completeness proof is along the lines given above, covers all possible attacks. Formalizing this argument is not trivial, since:

- not all instances of the variables occurring in a constraint system in solved form correspond to attacks; and
- to test the equivalence of two protocols we need to account for the equality tests the attacker can perform to analyze the responses of the honest agents.

We have bypassed the first difficulty by imposing that the attacker instantiates the first-order variables in a constraint system in solved form with intruder-generated constants, and proved that replacing these constants by any possible construction yields another attacks. This replacement is formalized by a well-founded ordering on the possible attacks, the ones in solved form being minimal for this ordering. For *finitary deduction systems* the set of minimal attacks is always finite. We solve the second difficulty by proving that it suffices to consider an attacker testing at most one equality on the result of his interaction with the honest process when this test is chosen *before* the computation of solved forms.

Simmetrically we account for disequalities in the process modeling the honest agents by introducing an equality chosen in a finite set before solving the constraint system.

Related works. Proofs of equivalence of processes are notoriously difficult, even when only finite processes are considered. In [13] Hüttel proves decidability for a fragment of the spi-calculus—*i.e.* for the Dolev-Yao model—for single trace processes. This result was extended by Baudet [14] in the same process setting but for subterm convergent equational

Labex CIMI, Toulouse

DOI 10.1109/PDP50117.2020.00074

theories. This result was extended in [8] for processes with several branches but no default case, *i.e.* without disequalities.

Another line of work starting from Hüttel focused on simple primitives but aimed at considering more complex processes. Among these we only consider those dealing with bounded processes, thereby misrepresenting ProVerif's [4] diff-equivalence by casting it in the world of bounded processes where it is too strong. An historical difficulty has been to find a satisfactory equivalence notion given a process algebra, *e.g.* applied-π calculus [15], spi-calculus [16], or psi-calculus [17]. It was latter proved in [18] that for deterministic processes labelled bisimilarity is equivalent to trace equivalence. This motivates us to shy away from complex process algebra and a translation to sets of traces, and define instead directly deterministic processes as sets of traces. In a similar setting but only for Dolev-Yao cryptographic primitives, [19] proved decidability of deterministic process equivalence.

In this paper we prove decidability of process equivalence for a class of deduction systems that at least [20] includes those with a subterm convergent equational theory as in [8] for processes that include disequalities as in [19]. Further work is needed to ellicit how large this class actually is.

Example finitary deduction systems. We remark that the standard Dolev-Yao deduction system [21] is finitary, since for every attack one can guess a subsequence of deduction steps which is itself an attack [22]. Deduction systems whose equational theory is subterm convergent are also finitary (see *e.g.* [20] though other articles considering these theories also include an argument that all attacks are representated).

Organization of this paper. Given this is a short version we have decided to provide a reference to the submitted version [23] and to only give in this paper a non-technical description in the hope that this will incite the brave reader to read the referenced version. Accordingly we follow the same organisation to ease the reading. Also we hope that this shorten format may help distinguish the original contribution that is not in the unpublished report [24].

We reuse in this paper the classical notions and notations for terms, equational theories, deduction systems recalled in Sec. II-A. We present the less known notion of symbolic derivation and finitary deduction systems in Sec. II-B. Equivalence of deterministic processes in our setting is defined in Sec. II-D and its decidability for finitary deduction systems is described first in the case of processes without disequalities, then in the general case. We conclude in Sec. III.

II. Procedure outline

This results builds on [24] in which a procedure computing whether two sequences of actions, called *symbolic derivations*, are equivalent assuming the observer is defined by a finitary deduction system. We refer to [23] for a longer version of this paper that includes the technical details necessary for comprehension.

A. Basic foundations

We represent values with terms describing how these can be constructed from a set of arbitrary values–the *constants*–with the application of functions. For example, using the well-known LISP list functions some values can be represented by the terms *nil*, $\mathrm{cons}(1,nil)$, $\mathrm{car}(\mathrm{cons}(1,nil))$. Traditionally the `car` function maps a couple to its first member. Thus the last value is also equal, more simply, to 1. The valid equalities are defined by an equational theory, a set of axioms of the form $l = r$ together with the axioms defining the equality as a congruence over values. In the preceding example we have assumed the axiom $\mathrm{car}(\mathrm{cons}(x,l)) = x$.

Unification is concerned with the resolution of equations entailed by an equational theory. A set of equations is *unifiable* if its variables can be instantiated by values such that all equations are entailed by the equational theory. One possible instantiation is called a *unifier*, and may contain variables. To proceed with the example, the mapping $\{x \mapsto \mathrm{cons}(y,z)\}$ is a unifier of the equation $\mathrm{car}(x) = y$ modulo the equational theory. It is even a *most general unifier* (mgu) of this equation: one can check that any unifier is an instance of this unifier. The existence of a set of a set of most general unifiers is not guaranteed. Even when such a set exists, it may be infinite. For the equational theory on word, $x \cdot \varepsilon = x, \varepsilon \cdot x = y, x \cdot (y \cdot z) = (x \cdot y) \cdot z$, the equation $a \cdot x = x \cdot a$ admits all the unifiers $\{x \mapsto a^n | n \in \mathbb{N}\}$, but this set cannot be described by the instances of a finite set of mgus. An equational theory is *finitary* if every set of equations has a finite (and possibly empty) set of mgus.

B. Symbolic derivations

We denote an execution of a process with symbolic derivations [25]. These are labeled sequences of values in which each value in the sequence is either in the initial memory of the process, received, copied from a preceding occurrence, constructed by the application of a function, or an arbitrary constant. Furthermore an occurrence of a value can be published, *i.e.* sent. A unification system records these constraints among the values.

Symbolic derivations are combined by identifying the published occurrences of one with the received occurrences of the other–and removing the reception label–to create a new symbolic derivation. In this model a realisable execution is modeled by a symbolic derivation in which all reception labels have been removed, called *closed* symbolic derivations and in which the unification system is satisfied by the values.

Among symbolic derivations we single out those defining the possible actions of an active attacker as those in which the initial knowledge only contains constants selected to represent random values. This setting is analoguous to the resolution of sets of equations modulo an equational theory. For satisfiability, a set of equation plays the same role wrt its unifiers that instantiante it to produce equations entailed by the equational theory as a symbolic derivations plays wrt to the attacker symbolic derivations that they can be combined with to form closed symbolic derivations whose unification system is satisfiable. This set of atacker symbolic derivations is called the set of solutions of the symbolic derivation.

There are however a few caveat to this analogy that are addressed in [23], [24]:

- Variables in sets of equations play the same role in this analogy as received values. However to ensure that the attacker indeed can interact with the symbolic derivation, we must ensure that the reception labels are removed. This problem is solved in [24], [25] by introducing an *opening* operation using which the constants in a solution can be re-used to allow for the combination with a new symbolic derivation;
- The solutions are quantified over all sequences of values *and* over all unification systems that represent tests that the attacker may apply on the results of his interaction with the symbolic derivation. This situation is resolved in [24], [25] by spliting attacker symbolic derivations in a pure deduction part, a *well-formed derivation* and a pure testing part, a *testing derivation*, that is latter connected using the opening operation.

The set of solutions of a symbolic derivation $\mathscr{C}$ is denoted $\mathscr{C}^{\star}$. The subset of well-formed solutions is denoted $\mathscr{C}^{\mathrm{sf}}$, and for a closed symbolic derivation $\mathscr{C}$. Two symbolic derivations are equivalent if and only if they have the same set of solutions.

A deduction system is defined by the set of operations the attacker can employ together with the equational theory defining the effects of these operations. We say that a deduction system $\mathscr{D}$ is *finitary* if for every symbolic derivation $\mathscr{C}$ the set $\mathscr{C}^{\star}$ can be described by a finite set of symbolic derivations $\min_{<}(\mathscr{C})$. The lack of technical details hides again two caveats:

- attacker symbolic derivations in $\min_{<}(\mathscr{C})$ also must be solutions of $\mathscr{C}$, for otherwise just instantiating the expected received values with constants would provide a meaningless description;
- also, since there is an unbounded number of constants, we need a pre-ordering on solutions to capture symbolic derivations that are instance one of the other (the strict ordering part) and one of another (the equivalence classes for the pre-ordering)

The resolution of these difficulties is presented in [23], [24].

C. Equivalence of processes

This part is novel and not present in [24]. We present in this section and in the next one how to reduce the problem of the equivalence of two processes to the problem of equivalence of two symbolic derivations, and how to take into account disequations as in [19] but for a more general class of deduction systems.

Since we assume finite processes, we identify processes with the set of symbolic derivations that correspont to finite forward executions. An *observational relation* is employed to restrict the symbolic derivations of the other process to which a symbolic derivation of one can potentially be identified. This relation is a pre-observational equivalence in the sense that any symbolic derivation in a process and any attacker trace which is in its solution set, there must exist a related trace in the other process which has that attacker trace in its solution set. Though it is superficially similar to the biprocess construction of [4] note that two different solutions of one symbolic derivation of one process can be solutions of two different symbolic derivations of the other process.

Finally we consider the containment problem of two processes, and define that two processes are equivalent if they are contained one in the other for the same observational relation R. The reduction of process equivalence to process containment is trivial.

Definition 1. *($\mathscr{D}$-Containment) Let P_1, P_2 be two processes and let R be an observational relation such that $R(P_1, P_2)$. We say that P_1 is $\mathscr{D}$-*contained in *P_2, and denote it $P_1 \sqsubseteq^R_{\mathscr{D}} P_2$, if $\mathscr{C}_1{}^{\star} \subseteq \cup_{1\leq i\leq n}\mathscr{C}_{2,i}{}^{\star}$ for all $\mathscr{C}_1 \in P_1$.*

D. Decidability of containment

1) Processes without disequalities: In the case of processes without disequalities, the procedure is straightforward. For each possible execution, we first guess an additional test that the attack will perform on his interaction with the execution ($\mathscr{C}_t$) and on which terms it will be performed ($\mathscr{C}_h \circ \mathscr{C}_t$). By [24] and if performed for all possible tests (one for each function), this test is sufficient to prove that all instances of a minimal solution of $\mathscr{C}_h$ are solutions of the other symbolic derivation. We have adapted it to prove that all solutions of $\mathscr{C}_h$ are solutions of one of the related symbolic derivation.

Algorithm 1 Algorithm for solving process containment.

procedure $\mathscr{D}$-PROCESS CONTAINMENT(P_1, P_2, R)
  Input: P_1, P_2 processes, R an observational relation
  Output: SAT *if* $(P_1, D_1) \sqsubseteq^R_{\mathscr{D}} (P_2, D_2)$.
  For all $\mathscr{C}_h \in P_1$
    For all $\mathscr{C}_t$ testing with one deduction and one equality
      let $\Sigma = \min_{<}(\mathscr{C}_h \circ \mathscr{C}_t)^{\mathrm{sf}, D_1(\mathscr{C}_h)}$
      For all $\mathscr{C} \in \Sigma$
        if there does not exist $\mathscr{C}'_h$ such that $R(\mathscr{C}_h, \mathscr{C}'_h)$ and $\mathscr{C} \in (\mathscr{C}'_h \circ \mathscr{C}_t)^{\star}$
        Then return UNSAT **End**
      End
    End
  End
  return SAT
end procedure

2) Processes with disequalities: Algorithm 2 is slightly more involved and adds a level of quantification. When testing a solution of a symbolic derivation $\mathscr{C}_h$ in P_1, we have to consider all related symbolic derivations $\mathscr{C}'_h$ in P_2 and for each guess a disequality that can be satisfied by an instance of the solution. Only if there is no such $\mathscr{C}'_h$ can we conclude that all instances of the solution are solutions of one of the related $\mathscr{C}'_h$.

III. CONCLUSION

We have introduced in this paper the notion of finitary deduction systems, and proved that symbolic equivalence is decidable for such attacker models. We believe that definition

Algorithm 2 Algorithm for solving extended process containment.

procedure $\mathscr{D}$-PROCESS CONTAINMENT(P_1, P_2, R)
Input: $\mathscr{P}_1 = (P_1, D_1), \mathscr{P}_2 = (P_2, D_2)$ *extended processes, R an observational relation*
Output: SAT *if* $(P_1, D_1) \sqsubseteq^R_{\mathscr{D}} (P_2, D_2)$.
For all $\mathscr{C}_h \in P_1$
For all $\mathscr{C}_t$ testing with one deduction and one equality
let $\Sigma = \min_<(\mathscr{C}_h \circ \mathscr{C}_t)^{\mathrm{sf}, D_1(\mathscr{C}_h)}$
For all $\mathscr{C} \in \Sigma$
If there does not exist $\mathscr{C}'_h$ such that $R(\mathscr{C}_h, \mathscr{C}'_h)$ and $\mathscr{C} \in (\mathscr{C}_h \circ \mathscr{C}_t)^\star$ and, for every $l \neq r \in D_2$, the HSD $(\mathrm{open}_C(\mathscr{C} \circ \mathscr{C}''_h \circ \mathscr{C}_t))^{\star, D_1} = \emptyset$ where: $\mathscr{C}''_h = \mathscr{C}'_h$ but for $\mathscr{S}''_h = \mathscr{S}'_h \cup \{l \stackrel{?}{=} r\}$
Then return UNSAT End
End
End
End
return SAT
end procedure

also captures the essence of lazy intruder techniques that are employed in many tools. In further work we aim at implementing this procedure to assess its practical feasability.

REFERENCES

[1] G. Lowe, "Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR," *Software - Concepts and Tools*, vol. 17, no. 3, pp. 93–102, 1996. [Online]. Available: http://web.comlab.ox.ac.uk/oucl/work/gavin.lowe/Security/Papers/NSFDR.ps

[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 *FMSE*, V. Shmatikov, Ed. ACM, 2008, pp. 1–10.

[3] S. Meier, B. Schmidt, C. Cremers, and D. A. Basin, "The TAMARIN prover for the symbolic analysis of security protocols," in *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, 2013, pp. 696–701.

[4] B. Blanchet, "Automatic verification of security protocols in the symbolic model: The verifier proverif," in *Foundations of Security Analysis and Design VII - FOSAD 2012/2013 Tutorial Lectures*, ser. Lecture Notes in Computer Science, A. Aldini, J. López, and F. Martinelli, Eds., vol. 8604. Springer, 2013, pp. 54–87. [Online]. Available: https://doi.org/10.1007/978-3-319-10082-1_3

[5] R. M. Amadio and D. Lugiez, "On the reachability problem in cryptographic protocols," in *CONCUR*, ser. Lecture Notes in Computer Science, C. Palamidessi, Ed., vol. 1877. Springer, 2000, pp. 380–394.

[6] J. K. Millen and V. Shmatikov, "Constraint solving for bounded-process cryptographic protocol analysis," in *ACM Conference on Computer and Communications Security*, 2001, pp. 166–175.

[7] M. Rusinowitch and M. Turuani, "Protocol insecurity with finite number of sessions is NP-complete," in *CSFW*. IEEE Computer Society, 2001, pp. 174–.

[8] R. Chadha, V. Cheval, Ştefan Ciobâcă, and S. Kremer, "Automated verification of equivalence properties of cryptographic protocols," *ACM Trans. Comput. Log.*, vol. 17, no. 4, pp. 23:1–23:32, 2016. [Online]. Available: http://dl.acm.org/citation.cfm?id=2926715

[9] M. Abadi and A. D. Gordon, "A calculus for cryptographic protocols: The spi calculus," in *ACM Conference on Computer and Communications Security*, 1997, pp. 36–47.

[10] M. Abadi and V. Cortier, "Deciding knowledge in security protocols under equational theories," in *ICALP*, ser. Lecture Notes in Computer Science, J. Díaz, J. Karhumäki, A. Lepistö, and D. Sannella, Eds., vol. 3142. Springer, 2004, pp. 46–58.

[11] Y. Chevalier, D. Lugiez, and M. Rusinowitch, "Towards an automatic analysis of web service security," in *Frontiers of Combining Systems, 6th International Symposium, FroCoS 2007, Liverpool, UK, September 10-12, 2007, Proceedings*, ser. Lecture Notes in Computer Science, B. Konev and F. Wolter, Eds., vol. 4720. Springer, 2007, pp. 133–147.

[12] M. Turuani, "The cl-atse protocol analyser," in *Term Rewriting and Applications, 17th International Conference, RTA 2006, Seattle, WA, USA, August 12-14, 2006, Proceedings*, ser. Lecture Notes in Computer Science, F. Pfenning, Ed., vol. 4098. Springer, 2006, pp. 277–286. [Online]. Available: https://doi.org/10.1007/11805618_21

[13] H. Hüttel, "Deciding framed bisimilarity," Jun. 2002, presented at the INFINITY'02 workshop.

[14] M. Baudet, "Deciding security of protocols against off-line guessing attacks," in *ACM Conference on Computer and Communications Security*, V. Atluri, C. Meadows, and A. Juels, Eds. ACM, 2005, pp. 16–25.

[15] S. Delaune, S. Kremer, and M. D. Ryan, "Symbolic bisimulation for the applied pi calculus," *Journal of Computer Security*, vol. 18, no. 2, pp. 317–377, 2010. [Online]. Available: https://doi.org/10.3233/JCS-2010-0363

[16] L. Durante, R. Sisto, and A. Valenzano, "Automatic testing equivalence verification of spi calculus specifications," *ACM Trans. Softw. Eng. Methodol.*, vol. 12, no. 2, pp. 222–284, 2003. [Online]. Available: https://doi.org/10.1145/941566.941570

[17] J. Borgström, "A complete symbolic bisimilarity for an extended spi calculus," *Electr. Notes Theor. Comput. Sci.*, vol. 242, no. 3, pp. 3–20, 2009. [Online]. Available: https://doi.org/10.1016/j.entcs.2009.07.078

[18] V. Cheval, V. Cortier, and S. Delaune, "Deciding equivalence-based properties using constraint solving," *Theor. Comput. Sci.*, vol. 492, pp. 1–39, 2013. [Online]. Available: https://doi.org/10.1016/j.tcs.2013.04.016

[19] V. Cheval, H. Comon-Lundh, and S. Delaune, "A procedure for deciding symbolic equivalence between sets of constraint systems," *Inf. Comput.*, vol. 255, pp. 94–125, 2017. [Online]. Available: https://doi.org/10.1016/j.ic.2017.05.004

[20] M. Kourjieh, "Logical Analysis and Verification of Cryptographic Protocols," Thèse de doctorat, Université Paul Sabatier, Toulouse, France, décembre 2009.

[21] D. Dolev and A. Yao, "On the Security of Public-Key Protocols," *IEEE Transactions on Information Theory*, vol. 2, no. 29, 1983.

[22] Y. Chevalier, D. Lugiez, and M. Rusinowitch, "Verifying cryptographic protocols with subterms constraints," in *LPAR*, ser. Lecture Notes in Computer Science, N. Dershowitz and A. Voronkov, Eds., vol. 4790. Springer, 2007, pp. 181–195.

[23] Y. Chevalier and B. F. Romero Jimenez, "Decidability of Deterministic Process Equivalence for Finitary Deduction Systems (short paper)," in *Euromicro International Conference on Parallel, Distributed and network-based Processing, Västerös, 11/03/2020-13/03/2020*, D. Masoud, F. Leporati, and S. Mikael, Eds. https://www.computer.org: IEEE Computer Society, mars 2020. [Online]. Available: https://oatao.univ-toulouse.fr/25220/1/main.pdf

[24] Y. Chevalier, "Finitary deduction systems," may 2011, https://arxiv.org/abs/1105.1376.

[25] Y. Chevalier and M. Rusinowitch, "Decidability of the equivalence of symbolic derivations," *Journal of Automated Reasoning.*, p. (to appear), Aug. 2010.