Using Task-Structured Probabilistic I/O Automata to Analyze Cryptographic Protocols

Ran Canetti[1], Ling Cheung[2], Dilsun Kaynar[3], Moses Liskov[4], Nancy Lynch[3], Olivier Pereira[5], and Roberto Segala[6]

[1] IBM T.J. Watson Center and Massachusetts Institute of Technology
[2] Radboud University of Nijmegen,
[3] Massachusetts Institute of Technology
[4] The College of William and Mary
[5] Université catholique de Louvain
[6] Università di Verona ⋆

Abstract. The Probabilistic I/O Automata (PIOA) framework of Lynch, Segala and Vaandrager provides tools for precisely specifying protocols and reasoning about their correctness based on implementation relationships between multiple levels of abstraction.
We enhance this framework to allow the analysis of protocols that use cryptographic primitives. For this purpose, we propose new techniques for handling nondeterministic behaviors, expressing computationally hardness assumptions, and for proving security in a composable setting.

1 Introduction

The task of modeling and analyzing of cryptographic protocols is typically complex, involving many subtleties and details, even when the analyzed protocols are simple. This causes security analysis of cryptographic protocols to be susceptible to errors and omissions (see [1–3] for instance). Our goal is to present a method for analyzing cryptographic protocols rigorously and systematically in a composable framework, while taking into account computational issues regarding cryptographic primitives.

This work is most closely related to the efforts of Backes, Pfitzmann and Waidner [4, 5], and of Lincoln, Mateus, Mitchell, Mitchell, Ramanathan, Scedrov and Teague [6, 7]. The main conceptual difference between these works and the current one lies in the way we handle nondeterminism, as we will see below.[1]

⋆ Canetti is supported by NSF CyberTrust Grant #430450; Cheung by DFG/NWO bilateral cooperation project 600.050.011.01 Validation of Stochastic Systems (VOSS); Kaynar and Lynch by DARPA/AFOSR MURI Award #F49620-02-1-0325, MURI AFOSR Award #SA2796PO 1-0000243658, NSF Awards #CCR-0326277 and #CCR-0121277, and USAF, AFRL Award #FA9550-04-1-0121; Pereira by the Belgian National Fund for Scientific Research (FNRS); and Segala by MURST project Constraint-based Verification of reactive systems (CoVer).

[1] A detailed comparison of our framework with the one of Backes & al. [4] is available in [8, 9].

2 Task-PIOAs

PIOAs Our approach is based on an extension of the Probabilistic I/O Automata (PIOA) framework developed in the concurrency semantics research community [10, 11]. Briefly, a PIOA is a kind of abstract automaton. It includes *states*, *start states*, *input*, *output*, and *internal actions*. Each action has an associated set of *transitions*, which go from states to probability distributions on states. PIOAs are capable of expressing both *probabilistic choices* and *nondeterministic choices*. PIOAs that model individual components of a system may be composed to yield a PIOA model for the entire system.

Task-PIOAs Traditionally, centralized, perfect-information schedulers are used to resolve nondeterministic choices in a PIOA. Such a scheduler has full knowledge about the past execution and is too powerful for computational analysis of security protocols: it might provide covert channels by scheduling actions of adversarial components as a function of secrets of protocol parties. To address this issue, we propose a distinction between *high-* and *low-level* nondeterminism. High-level nondeterminism refers to the adversarially observable events, for instance, message transmission on the network. This type of nondeterminism, which is standard in the cryptographic community, is algorithmically resolved by the automaton representing the adversary. Low-level nondeterminism refers to the ordering of events that are not controlled by adversarial components, e.g., internal or output transitions of protocol parties. As observed in the concurrency community, this type of nondeterminism is quite useful in protocol specification: by leaving the ordering of events unspecified whenever possible, we reduce the amount of inessential details (the so-called "clutters") contained in our models. This often simplifies correctness proofs involving implementation relations. More importantly, the resulting correctness statement is more general, because it is valid no matter how the nondeterministic choices are resolved in a real-life implementation. Thus, an implementer has more freedom to make design decisions based on the specific context in which the protocol is used (e.g., hardware and network characteristics). We think that capturing and separating these two types of nondeterminisms is an aspect in which our approach differs from all existing cryptographic frameworks.

To resolve low-level nondeterminism in a consistent manner, we extend the PIOA framework with a new *task* mechanism, obtaining task-PIOAs as detailed in [8]. Basically, a task-PIOA is a pair $(\mathcal{P}, R)$ where $\mathcal{P}$ is a classical PIOA and R is an equivalence relation on internal and output actions of $\mathcal{P}$. Each equivalence class of R is called a task, and we require that tasks satisfy the *action determinism* axiom, which states that, given a state and a task of a task-PIOA, there is at most one (probabilistic) action of that task that is enabled in that state. We use tasks to abstract from the actual state variables of the protocol parties: for instance, a task will be "send first protocol message", without reference to a specific message content. Given this task mechanism, we define schedulers as simply an arbitrary sequence of tasks, called a task scheduler. The

action determinism property of tasks guarantees that specifying a task scheduler for a given task-PIOA resolves all nondeterministic choices and defines a purely probabilistic execution.

Time-bounded Task-PIOAs The security of cryptographic protocols typically relies on the assumption that certain problems cannot be solved with nonnegligible probability by resource bounded entities. In order to capture these bounds, we define *time-bounded* task-PIOAs [9]. Basically, a task-PIOA $\mathcal{T}$ is b-time-bounded if, assuming a standard bit representation, (i) all its components (state, actions, tasks, ...) can be represented as bit strings of length at most b, (ii) there is a Turing machine that can decide in time at most b if a bit string is the representation of a task-PIOA component (iii) given a task and a state, there is a Turing machine that can determine in time at most b the unique enabled action (if there is one), (iv) given a state and an action, there is a Turing machine that can compute in time at most b the next state of $\mathcal{T}$. Furthermore, we require that all these Turing machines can be described as bit strings of length at most b, given some standard encoding.

Typically, a computational hardness assumption states that, as the size of a problem grows, the success probability of a resource-bounded entity trying to solve the problem diminishes quickly. The size of a problem is expressed in terms of a *security parameter* $k \in \mathsf{N}$. Accordingly, we define families of task-PIOAs indexed by a security parameter: a *task-PIOA family* $\overline{\mathcal{T}}$ is an indexed set $\{\mathcal{T}_k\}_{k\in\mathsf{N}}$ of task-PIOAs. The notion of time bound is also expressed in terms of the security parameter; namely, given $b : \mathsf{N} \to \mathsf{R}^{\geq 0}$, we say that $\overline{\mathcal{T}}$ is *b-bounded* if every $\mathcal{T}_k$ is $b(k)$ time-bounded. Also, we say that a family $\overline{\mathcal{T}}$ is *polynomial-time-bounded* if it is bounded by a polynomial function.

3 Proving Security

Defining Security We perform security analysis along the lines of Universally Composable Security [12] and Reactive Simulatability [5]. Following these approaches, the functionality to be achieved by a protocol is described by a task-PIOA family $\overline{\mathcal{F}}$, which typically models a kind of trusted party that computes the correct result from given inputs. A protocol $\overline{\mathcal{P}}$ is defined to be secure if, for any adversary $\overline{\mathcal{A}}$ that interacts with the protocol, there exists a "simulator" $\overline{\mathcal{S}}$ that interacts with the functionality such that no external environment can distinguish whether it is interacting with the protocol and $\overline{\mathcal{A}}$ or, with the functionality and $\overline{\mathcal{S}}$. In the task-PIOA framework we express this indistinguishability notion by saying that the composition of $\overline{\mathcal{P}}$ and $\overline{\mathcal{A}}$ must implement the composition of $\overline{\mathcal{F}}$ and $\overline{\mathcal{S}}$, which we denote by $\overline{\mathcal{P}}\|\overline{\mathcal{A}} \leq_{neg,pt} \overline{\mathcal{F}}\|\overline{\mathcal{S}}$. More precisely, the $\leq_{neg,pt}$ implementation relation means that, for every polynomial time-bounded environment family $\overline{\mathcal{E}}$ and every polynomial length-bounded task scheduler for $\overline{\mathcal{P}}\|\overline{\mathcal{A}}\|\overline{\mathcal{E}}$, there is a polynomial length-bounded task scheduler for $\overline{\mathcal{F}}\|\overline{\mathcal{S}}\|\overline{\mathcal{E}}$ such that the probabilities that $\overline{\mathcal{E}}$ performs an *accept* output actions in these two systems differ by a negligible amount. An important part of this definition is

that it includes the quantifiers on the task-schedulers, that is, we prove security properties for every way to resolve the low-level nondeterminism. In [9], we prove convenient properties of the $\leq_{neg,pt}$ relation; for example, it is transitive and is preserved when we compose two $\leq_{neg,pt}$-related systems with any polynomial-time-bounded task-PIOA family.

Proving Security In order to prove that two task-PIOA families $\overline{\mathcal{A}}$ and $\overline{\mathcal{B}}$ are $\leq_{neg,pt}$ related, we decompose our security proofs in several steps, as in game-based proofs [13, 14]: in order to prove that $\overline{\mathcal{A}} \leq_{neg,pt} \overline{\mathcal{B}}$, we prove that $\overline{\mathcal{A}}_1 \leq_{neg,pt} \cdots \leq_{neg,pt} \overline{\mathcal{A}}_n$, where $\overline{\mathcal{A}}_1 = \overline{\mathcal{A}}$ and $\overline{\mathcal{A}}_n = \overline{\mathcal{B}}$. The families $\overline{\mathcal{A}}_i$ and $\overline{\mathcal{A}}_{i+1}$ are defined in such a way that, either they are perfectly indistinguishable, that is, indistinguishable even by an unbounded environment, or they only differ by a small detail corresponding to a computational assumption.

When $\overline{\mathcal{A}}_i$ and $\overline{\mathcal{A}}_{i+1}$ are perfectly indistinguishable, we prove that $\overline{\mathcal{A}}_i \leq_{neg,pt} \overline{\mathcal{A}}_{i+1}$ by using a new, sound, simulation relation [8, 9]. Even though the form of this simulation relation is not usual (for instance, it relates probability distribution on executions rather than states), this type of proof requires using fairly traditional formal methods.

In order to relate systems that are indistinguishable in a computational sense only, we translate computational hardness assumptions in terms of implementation relations between task-PIOA families: for instance, for expressing the DDH assumption, we define a family $\overline{\mathcal{DDH}}_1$ transmitting a triple (g^x, g^y, g^{xy}), a family $\overline{\mathcal{DDH}}_2$ transmitting a triple (g^x, g^y, g^z) (where x, y, and z are selected randomly), and claim that $\overline{\mathcal{DDH}}_1 \leq_{neg,pt} \overline{\mathcal{DDH}}_2$. In [9], we prove for a similar case that this formulation style is equivalent to the classical computational definition.

In order to exploit these computational assumptions, we define $\overline{\mathcal{A}}_i$ and $\overline{\mathcal{A}}_{i+1}$ in such a way that they can be expressed as $\overline{\mathcal{C}}_1 \| \overline{Ifc}$ and $\overline{\mathcal{C}}_2 \| \overline{Ifc}$ respectively, where $\overline{\mathcal{C}}_1 \leq_{neg,pt} \overline{\mathcal{C}}_2$ is a stated computational assumption and $\overline{Ifc}$ is polynomial-time-bounded (the $\overline{Ifc}$ family plays the role of the reduction in classical computational proofs). Now, the relation $\overline{\mathcal{A}}_i \leq_{neg,pt} \overline{\mathcal{A}}_{i+1}$ follows from the composition property of the $\leq_{neg,pt}$ relation.

As a case-study, we used these techniques in [9] for the analysis of a classical Oblivious Transfer protocol [15]. This analysis involves a passive adversary, as the OT protocol we consider is only secure in front of this type of opponent. This passive adversary is modeled by restricting the task-PIOA describing the adversary to only send messages he previously received. Dealing with an active adversary would simply correspond to removing this restriction in the adversary definition.

4 Conclusion

When working on our OT case-study, we found that breaking down our proofs into several pieces in order to separate issues of probability from computational issues was specially convenient: probability issues can be dealt with using fairly

traditional formal methods, while computational issues are concentrated on isolated pieces, which can be managed independently with traditional cryptographic techniques: formulating computational assumptions, and building reductions. This should provide a way for people from the formal and computational cryptography communities to work together on proofs in a single framework.

We used our task-PIOA framework to establish composable security proofs. So, we expect that it can eventually be used to obtain sound abstractions for classical cryptographic primitives and protocols. This would allow performing sound, symbolic-style, analysis inside our framework: symbolic analysis of cryptographic protocols based on I/O automata has already been performed in [16].

Our plans for the near future include establishing general composition theorems in the style of [12, 5], to model more sophisticate computational assumptions, and to use these results for the analysis of a key exchange protocol.

References

1. Choo, K.K.R., Boyd, C., Hitchcock, Y.: Errors in computational complexity proofs for protocols. In: Advances in Cryptology - Asiacrypt 2005. Volume 3788 of LNCS., Springer (2005) 624–643
2. Hofheinz, D., Müller-Quade, J., Steinwandt, R.: Initiator-resilient universally composable key exchange. In Snekkenes, E., Gollmann, D., eds.: European Symposium on Research in Computer Security, Proceedings of ESORICS 2003. Volume 2808 of LNCS., Springer (2003) 61–84 Full version available on `http://eprint.iacr.org/2003/063/`.
3. Shoup, V.: OAEP reconsidered. In: Advances in Cryptology – CRYPTO 2001. Volume 2139 of LNCS., Springer (2001) 239–259
4. Backes, M., Pfitzmann, B., Waidner, M.: Secure asynchronous reactive systems. Cryptology ePrint Archive, Report 2004/082 (2004) `http://eprint.iacr.org/`.
5. Pfitzmann, B., Waidner, M.: A model for asynchronous reactive systems and its application to secure message transmission. In: IEEE Symposium on Security and Privacy, Oakland, CA, IEEE Computer Society (2001) 184–200
6. Lincoln, P., Mitchell, J., Mitchell, M., Scedrov, A.: A probabilistic poly-time framework for protocol analysis. In: Proceedings of the 5th ACM conference on Computer and communications security (CCS-5), San Francisco (1998) 112–121
7. Ramanathan, A., Mitchell, J., Scedrov, A., Teague, V.: A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. Theoretical Computer Science **353** (2006) 118–164
8. Canetti, R., Cheung, L., Kaynar, D., Liskov, M., Lynch, N., Pereira, O., Segala, R.: Task-structured probabilistic I/O automata. MIT CSAIL Technical Reports - MIT-CSAIL-TR-2006-023 (2006) 45 pages.
9. Canetti, R., Cheung, L., Kaynar, D., Liskov, M., Lynch, N., Pereira, O., Segala, R.: Using task-structured probabilistic I/O automata to analyze an oblivious transfer protocol. MIT CSAIL Technical Reports - MIT-CSAIL-TR-2006-019 (2006) 98 pages.
10. Lynch, N., Segala, R., Vaandrager, F.: Compositionality for probabilistic automata. In Amadio, R., Lugiez, D., eds.: Proceedings of the 14th International Conference on Concurrency Theory, CONCUR'03, Marseille, France, Springer-Verlag - LNCS

Vol. 2761 (2003) 208–221 Fuller version appears in Technical Report MIT-LCS-TR-907, Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA.
11. Segala, R.: Modeling and Verification of Randomized Distributed Real-Time Systems. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA (1995) Also, MIT/LCS/TR-676.
12. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In Naor, M., ed.: Proceedings of the 42nd Annual Symposium on Foundations of Computer Science, IEEE Computer Society (2001) 136–145 Full version available on `http://eprint.iacr.org/2000/067`.
13. Bellare, M., Rogaway, P.: The game-playing technique and its application to triple encryption. Cryptology ePrint Archive, Report 2004/331 (2004) `http://eprint.iacr.org/`.
14. Shoup, V.: Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004) `http://eprint.iacr.org/`.
15. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game - a completeness theorem for protocols with honest majority. In: Proceedings of the 19th Annual ACM Symposium on the Theory of Computing (STOC), ACM Press (1987) 218–229
16. Lynch, N.: I/O automaton models and proofs for shared-key communication systems. In: 12th IEEE Computer Security Foundations Workshop — CSFW'99, Mordano, Italy, IEEE Computer Society Press (1999) 14–29