Computer-aided cryptographic proofs

Gilles Barthe[1], Benjamin Grégoire[2], and Santiago Zanella Béguelin[3]

[1] IMDEA Software Institute
[2] INRIA Sophia Antipolis - Méditerranée
[3] Microsoft Research

Provable security [6] is at the heart of modern cryptography. It advocates a mathematical approach in which the security of new cryptographic constructions is defined rigorously, and provably reduced to one or several assumptions, such as the hardness of a computational problem, or the existence of an ideal functionality. A typical provable security statement is of the form: for all adversary $\mathcal{A}$ against the cryptographic construction $\mathcal{S}$, there exists an adversary $\mathcal{B}$ against a security assumption $\mathcal{H}$, such that if $\mathcal{A}$ has a high probability of breaking the scheme $\mathcal{S}$ in time t, then $\mathcal{B}$ has a high probability of breaking the assumption $\mathcal{H}$ in time t' (defined as a function of t).

EasyCrypt [1] is a framework for building and verifying machine-checked security proofs for cryptographic constructions in the computational model. Following the code-based approach [4], EasyCrypt uses probabilistic programs with adversarial computations to formulate unambiguously reductionist arguments. In EasyCrypt, cryptographic constructions are modelled as probabilistic programs, and their security is given by the probability of an event in a experiment, where an adversary interacts with the construction; similarly, security assumptions are stated in terms of the probability of an event in a probabilistic experiment. The key novelty of EasyCrypt (and its predecessor CertiCrypt [2]) is to provide programming languages tools to capture common reasoning patterns in cryptographic proofs. In particular, EasyCrypt provides support for a probabilistic relational Hoare Logic (pRHL) [2], whose judgments $\models c_1 \sim c_2 : \Psi \Rightarrow \Phi$ relate two probabilistic programs c_1 and c_2 (that typically involve adversarial code) relative to a pre-condition Ψ and a post-condition Φ, both defined as relations over program states. Informally, a judgment is valid iff for every initial memories that are related by the pre-condition, the sub-distributions of final memories are related by the lifting of the post-condition to distributions; the definition of the lifting operator $\mathcal{L}$ is adopted from probabilistic process algebra [7], and has close connections with the Kantorovich metric, and with flow networks [5]. As security properties are typically expressed in terms of probability of events rather than pRHL judgments, EasyCrypt implements mechanisms to derive from valid judgments probability claims, i.e. inequalities between expressions of the form $\Pr[c, m : S]$ that denote the probability of the event S in the sub-distribution $[\![c]\!]\ m$.

To automate reasoning in pRHL, EasyCrypt implements an automated procedure that given a logical judgment involving loop-free closed programs, computes a set of sufficient conditions for its validity, known as verification conditions. In the presence of loops or adversarial code, we require the user to provide the necessary annotations. The outstanding feature of this procedure, and the key to its

effectiveness, is that verification conditions are expressed as first-order formulae, without any mention of probability, and thus can be discharged automatically using off-the-shelf SMT solvers and theorem provers.

To date, EasyCrypt (and its predecessor CertiCrypt) have been used to verify prominent examples of cryptographic constructions, including the OAEP padding scheme, the Cramer-Shoup encryption scheme, the Full Domain Hash signature scheme, the Merkle-Damgård hash function design, and zero-knowledge proofs. Moreover, CertiCrypt and EasyCrypt have been extended to reason about differentially private computations [3]. More recently, EasyCrypt has been used for the first time to prove the security of a novel cryptographic construction. Specifically, we have used EasyCrypt to prove the IND-CCA security of ZAEP, a redundancy-free public-key encryption scheme based on the Rabin function and RSA with exponent 3.

More information about the project can be found at:

`http://easycrypt.gforge.inria.fr`

References

1. Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. Computer-aided security proofs for the working cryptographer. In *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 71–90, Heidelberg, 2011. Springer.
2. Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. Formal certification of code-based cryptographic proofs. In *36th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009*, pages 90–101, New York, 2009. ACM.
3. Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. Probabilistic reasoning for differential privacy. In *39th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2012*, pages 97–110, New York, 2012. ACM.
4. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 409–426, Heidelberg, 2006. Springer.
5. Yuxin Deng and Wenjie Du. Logical, metric, and algorithmic characterisations of probabilistic bisimulation. Technical Report CMU-CS-11-110, Carnegie Mellon University, March 2011.
6. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.
7. Bengt Jonsson, Wang Yi, and Kim G. Larsen. Probabilistic extensions of process algebras. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of Process Algebra*, pages 685–710. Elsevier, Amsterdam, 2001.