(Extended Abstract - Submitted to the 2nd Informal ACM SIGPLAN Workshop on Mechanizing Metatheory, WMM 2007)

A Framework for Language-Based Cryptographic Proofs

Gilles Barthe Benjamin Grégoire Romain Janvier Santiago Zanella Béguelin

INRIA Sophia-Antipolis, France Microsoft Research - INRIA Joint Centre, France

{Gilles.Barthe,Benjamin.Gregoire,Romain.Janvier,Santiago.Zanella} @sophia.inria.fr

Motivation In cryptography, provable security advocates a mathematical approach where the goals and requirements of cryptographic systems are specified precisely, and where the security proof is carried out rigorously and makes explicit the assumptions it relies upon. Typically, security objectives are expressed in complexity-theoretical terms and refer to the probability of an efficient adversary to thwart a security objective (e.g. secrecy), whereas security proofs are "reductionist", i.e. prove that the existence of an efficient adversary contradicts a computational assumption (e.g. that the Decisional Diffie-Hellman problem is hard).

The game-playing technique is a general method to structure and unify cryptographic proofs that has been widely applied in the literature. In essence, the game-playing technique suggests to view the interaction between an unknown efficient adversary and a cryptosystem as a probabilistic game depending on security a parameter η where the winning probability of the adversary corresponds to the probability of breaking a given security property. The initial game is stepwise transformed in a security-preserving fashion into a final game where it is easy to analyze and bound the winning probability. Since the transformations are security-preserving, one can argue that the same bound holds for the initial game and, if this bound is a negligible function of η, then the probability of breaking the security property of the system is also negligible.

Although the adoption of provable security and the game-playing technique has significantly enhanced confidence in cryptographic systems, the community is increasingly wary about security proofs: several published proofs have been found incorrect, and in general proofs are becoming too complex to be verified. This is partly due to the fact that proofs are rather involved and rely on different kinds of mathematical reasoning including complexity theory, probability theory and group theory. However, the main reason is to be found in the difficulty in pinpointing the underlying hypotheses in the proof and in isolating the creative and original parts from the uninteresting steps recurring – with variations – in every other proof.

Bellare and Rogaway [1], and Halevi [3] propose the game-playing technique as a natural solution for taming the complexity of proofs and recognize that a fully-specified programming language is required to code games. We believe cryptographers could greatly benefit from a framework for formalizing and verifying the transformations in game-based cryptographic proofs and propose a language-based framework built on top of the Coq proof assistant.

The framework We define a simple WHILE language augmented with a random sampling instruction and procedure calls and give a monadic continuation-passing style probabilistic semantics. The language is expressive enough to code the games appearing in the literature.

Transformations found in game-based proofs fall in three main classes: 1) Semantic-preserving transformations; 2) Transformation based on indistinguishability; 3) Transformations that increase the winning probability of the adversary.

The first class includes common compiler optimizations like code motion, dead code elimination, common subexpression elimination, branch predicting, constant propagation and procedure inlining. For each of these optimizations we have developed Coq tactics based on reflection that perform these optimizations (i.e. they allow to substitute a goal of the form $P_1 \equiv P_2$ for $P_1 \equiv P_2'$ for some P_2' identical to P_2 up to optimizations). These tactics are proved sound w.r.t the semantics.

The second class refers to transformations where the chance of distinguishing the probability distributions generated by two games is negligible. For example, since the probability of uniformly sampling a particular bitstring of length η is negligible, these two games are identical up to a negligible factor:

$$x \overset{\$}{\leftarrow} \{0,1\}^\eta; \mathbf{if}\ x = y\ \mathbf{then}\ P_1\ \mathbf{else}\ P_2 \simeq_{1/2^\eta} x \overset{\$}{\leftarrow} \{0,1\}^\eta; P_2$$

The third-class of transformations is usually used to transform a game into a simpler one whenever finding a tight bound is not the objective. These last two classes of transformations are currently supported in the framework by a library of ad hoc lemmas but we believe they can be automated in some scenarios.

The contribution We know of only one other tool to assist proofs in the computational model of cryptography, *CryptoVerif* [2]. However, it differs fundamentally to ours: *CryptoVerif* performs an heuristic-based search on a library of (user-provided and predefined) transformations to generate a sequence of games for a proof and gives little attention to whether the transformations are computationally sound, while our framework relies on the user to supply the sequence of games but instead put the emphasis on verifying that the whole proof is semantically correct. We believe the two approaches are complementary and can benefit from each other.

References

[1] Mihir Bellare and Phillip Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint Archive, Report 2004/331, 2004. `http://eprint.iacr.org/`.

[2] Bruno Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 2007. Special issue IEEE Symposium on Security and Privacy 2006. To appear.

[3] Shai Halevi. A plausible approach to computer-aided cryptographic proofs. Cryptology ePrint Archive, Report 2005/181, 2005. `http://eprint.iacr.org/`.