

From CryptoVerif Specifications to Computationally Secure Implementations of Protocols

(Work in Progress)

David Cadé

École Normale Supérieure, CNRS, INRIA

CryptoVerif [Bla08] is a protocol verifier in the computational model that can automatically prove properties of protocols, such as secrecy and authentication. It generates proofs by sequences of games, like those written manually by cryptographers. The first game describes the specification of the protocol to prove; the next games are deduced by relying on security assumptions on primitives or by syntactic transformations. These transformations are such that consecutive games are indistinguishable and, in the last game, the desired security property is obvious. So the first game also satisfies the property. The games are formalized in a probabilistic polynomial-time process calculus.

We are implementing a compiler that takes a CryptoVerif specification and generates an implementation of the protocol in OCaml. The goal of this work is to obtain implementations of security protocols proved secure in the computational model. We will prove that our compiler is correct, that is, the semantics of the generated code corresponds to the semantics of the specification. Therefore, if CryptoVerif can prove a property on the protocol, then the implementation will also satisfy this property.

The CryptoVerif specification is annotated by the user with hints about the implementation details: the user indicates how the specification is split into modules (*e.g.*, the key generator part, the client part, and the server part), the files that will store data shared between several modules, the name of the OCaml functions corresponding to the cryptographic primitives, and the size of the random numbers. The compiler generates code for each module. For each step of the protocol, inputting a message and outputting the response, the compiler generates a function that takes the received message as argument and returns the response. These functions can then be called by a manually implemented network layer, which can be considered as part of the adversary, so we need not make any security assumptions on it. On the other hand, the CryptoVerif specification makes security assumptions on the cryptographic primitives. We do not verify that the OCaml implementation of these primitives satisfies these assumptions: they still need to be proved manually, but the CryptoVerif speci-

fication states precisely what has to be proved.

A related approach is the work of Bhargavan et al. about the verification of protocols written in ML [BCF07, BCFZ09] and its application to TLS [BCFZ08]. They generate a CryptoVerif specification from a ML implementation and then verify it with CryptoVerif. In contrast to their work, we generate a ML implementation from a CryptoVerif specification. Their use of ML as input language, which is more flexible than the CryptoVerif specification language and also better known, is an advantage of their approach. However they are still limited by the expressive power of CryptoVerif: one sometimes has to tweak the specification in order to prove the desired security properties. Our approach makes it easier to write a specification well-suited for CryptoVerif. Our approach also favors the methodology of first writing the formal specification, proving it and second implementing it. Our compiler facilitates this second step. Finally, we separate clearly what part of the implementation to prove manually (the primitives), what is proved automatically by CryptoVerif (the protocol), and what does not need any proof (the network code). This manual separation simplifies the verification process.

Acknowledgments

We would like to thank Bruno Blanchet for his help on this work. This work was partly supported by the ANR project FormaCrypt and by DGA.

References

- [BCF07] Karthikeyan Bhargavan, Ricardo Corin, and Cédric Fournet. Crypto-verifying protocol implementations in ML. In *Workshop on Formal and Computational Cryptography (FCC'07)*, Venice, Italy, July 2007.
- [BCFZ08] Karthikeyan Bhargavan, Ricardo Corin, Cédric Fournet, and Eugen Zălinescu. Cryptographically verified implementations for TLS. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*, pages 459–468. ACM, October 2008.
- [BCFZ09] Karthikeyan Bhargavan, Ricardo Corin, Cédric Fournet, and Eugen Zălinescu. Automated computational verification for cryptographic protocol implementations. Unpublished draft available at <http://www.msr-inria.inria.fr/projects/sec/fs2cv/>, 2009.
- [Bla08] Bruno Blanchet. A computationally sound mechanized prover for security protocols. *IEEE Transactions on Dependable and Secure Computing*, 5(4):193–207, October–December 2008.