

Cryptographically Verified Implementations for TLS

Karthikeyan Bhargavan
Microsoft Research

Ricardo Corin
MSR-INRIA Joint Centre

Cédric Fournet
Microsoft Research

Eugen Zălinescu
MSR-INRIA Joint Centre

February 11, 2009

Recent advances in formal methods and tools enable the automated verification of complex security protocols. However, these tools remain difficult to apply, since verification occurs independently of the development process, rather than during design, prototyping, and testing. We are thus interested in integrating protocol verifiers to these phases, by narrowing the gap between concrete implementations and verified models.

We extract symbolic and computational models from executable code, relying in particular on a new tool for extracting CryptoVerif [1] scripts from ML. We share as much code as possible between implementations and models: they differ mostly in their implementations of core cryptographic libraries, either as concrete code or as encoded assumptions on the adversary.

As a case study, we consider the Transport Layer Security protocol (TLS) [2], one of the most widely deployed communications protocol. We program a small functional implementation of TLS 1.0 in ML. Both client and server code interoperate with mainstream implementations. We obtain a range of positive security results, covering both symbolic and computational cryptographic aspects of the protocol. More details can be found in [3].

Acknowledgements We thank Bruno Blanchet and Bogdan Warinschi for helpful discussions on computational verification during this work.

References

- [1] Bruno Blanchet. Computationally sound mechanized proofs of correspondence assertions. In *20th IEEE Computer Security Foundations Symposium (CSF'07)*, pages 97–111, 2007.
- [2] T. Dierks and C. Allen. The TLS protocol version 1.0. RFC 2246, IETF, 1999.
- [3] <http://www.msr-inria.inria.fr/projects/sec/fs2cv/index.html>.