ARA SSIA FormaCrypt (2005)

Participants: LIENS B. Blanchet, D. Cadé, D. Monniaux, D. Pointcheval
LSV J. Goubault-Larrecq, M. Baudet, H. Comon-Lundh, S. Delaune, S. Kremer, L. Mazaré
LORIA V. Cortier, H. Hördegen, M. Turuani, B. Warinschi, E. Zalinescu
Scientific advisor: M. Abadi

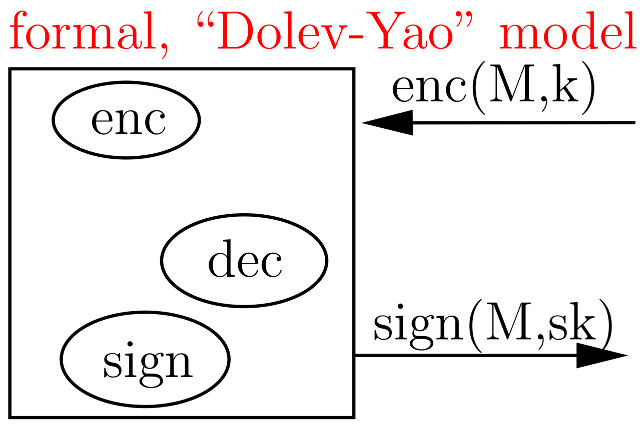
Contact: Bruno Blanchet, `blanchet@di.ens.fr`

URL: `http://www.di.ens.fr/~blanchet/formacrypt/`

Objectives of the project

Two models for the verification of cryptographic protocols:

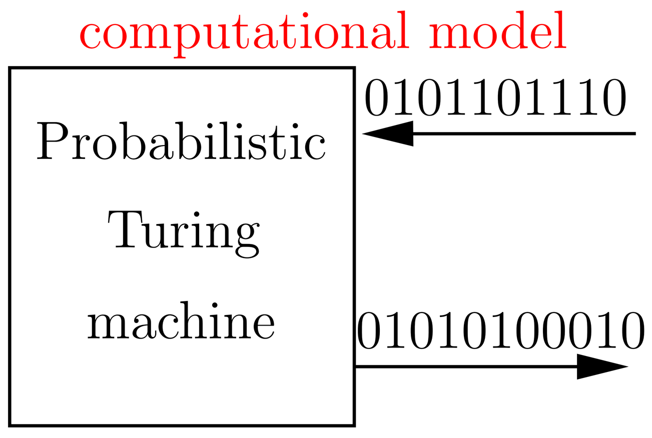
formal, "Dolev-Yao" model



abstract model; automatic proofs

Our goal:
bridge the gap between these two models

computational model



realistic model; manual proofs

A computationally sound prover

Goal:
Build a specialized, computationally sound, automatic prover.

Results:
An automatic, computationally sound prover, CryptoVerif, that

- generates proofs by sequences of games, as in Shoup's or Bellare and Rogaway's method;
- proves secrecy and correspondence assertions (authentication);
- provides a generic treatment of cryptographic primitives, including shared- and public-key encryption, signatures, MACs, hash functions, computational Diffie-Hellman;
- is sound in the presence of an active adversary, for a parametric number of sessions;
- evaluates the probability of an attack (exact security).

The user is allowed (but does not have) to interact with the prover to make it follow a specific sequence of games.

CryptoVerif is available at `http://www.cryptoverif.ens.fr/`.

Examples handled:

- many protocols: correct versions of Needham-Schroeder, Denning-Sacco, Otway-Rees, Yahalom, . . . protocols;
- Full Domain Hash signature scheme;
- encryption schemes of Bellare and Rogaway, CCS'93;
- Kerberos, with and without PKINIT.

Planed extensions:

- Other primitives, such as decisional Diffie-Hellman, xor.
- Additional game transformations.

A computationally sound logic

Goal:
Design a computationally sound logic for reasoning symbolically on protocols.

Results:

- Adaptation of the Protocol Composition Logic (PCL) to the computational model.
  Soundness proof for a subset of PCL with positive tests.
- Extension to prove more complex properties, such as secrecy of keys.

This logic is compositional. For example, from the security of keys established using a key exchange protocol, one can prove the security of a secure channel application that uses these keys.

The modular approach

Goal:
Obtain computational soundness results, *i.e.*, show that security in the formal model implies security in the computational model.

Results:

- Computational soundness was shown for public-key encryption and signatures.
  Based on this result, we have implemented a tool that provides computational proofs of protocols, using the AVISPA formal protocol analyzer, available at `http://www.avispa-project.org/`.
- We have extended computational soundness results to the case of hash functions, with a stronger notion of symbolic secrecy, decidable for a bounded number of sessions.
- For symmetric encryption, computational soundness typically requires the absence of key cycles. We have shown that this property is decidable for a bounded number of sessions.
- We have developed an equational theory for specifying cryptographic primitives, such that (symbolic) static equivalence is sound with respect to computational indistinguishability.
  This result includes the possibility for an adversary to guess low entropy values, such as passwords (guessing attacks).
- We have shown the first soundness results for observational equivalence which allows to prove general indistinguishability properties in the presence of an active adversary.

Planed extensions:

- Branching properties (*e.g.*, fairness).
- Primitives with more complex equational theories (Diffie-Hellman, XOR, CBC encryption).
- Modular proof techniques allowing to extend the protocol language and to combine soundness results.

Case studies and comparison of the various approaches

Goal: Compare the results obtained by these approaches.
Result: Comparison between two analyses of the Wide-Mouth-Frog protocol, one by ProVerif and a computational soundness theorem, one by CryptoVerif.