# Verification of security protocols
# – Squirrel prover –

**Laboratory, institution and university.** Univ Rennes, CNRS, IRISA, France & LSV (ENS Paris-Saclay). The internship will be located at IRISA, Rennes and/or at LSV (ENS Paris-Saclay), Gif-sur-Yvette.

**Team or project of the Lab.** EMSEC team at IRISA and SECSI team at LSV.

**Name and email address of the advisor.**
David Baelde, `baelde@lsv.fr`, and Stéphanie Delaune `stephanie.delaune@irisa.fr`

**Indemnisation.** The internship is supported by the European grant POPSTAR (ERC Starting Grant) and the ANR grant TECAP.

**Context.** Security protocols are distributed programs that aim at securing all kinds of communications, by using cryptography to ensure security properties, such as confidentiality, authentication or anonymity. Such protocols are widely deployed, *e.g.* for electronic commerce on the Internet, in banking networks, mobile phones and more recently online elections. As the goal is to provide security guarantees even when communicating over an untrusted network – typically the Internet – it turns out that these protocols are extremely difficult to get right. Formal methods have proved to be a very useful tool to detect errors, and verify the correctness of security protocols. Traditionally two approaches have been used: the *computational* one, which captures strong notions of security and offers guarantees against all probabilistic polynomial-time attackers, and the *symbolic* one in which things are modelled more abstractly and which is more amenable to automation. To get an idea of this research area, the interested reader can consult [CK11].

A few years ago, Bana and Comon proposed a new approach to security proofs [BC14]. This approach, which they call *computationally complete symbolic attacker* (CCSA), uses the symbolic formal setting of first-order logic, but avoids the limitations of the symbolic model. This approach has been demonstrated on various protocols to obtain formal proofs of security, e.g. [CK17]. There are however two limitations that prevent a more widespread use of this approach. First, it is limited to bounded executions. Second, proofs are manual: proving a non-trivial goal in detail is tedious; proving all the goals resulting from all the possible executions is not manageable. To overcome these limitations, a framework composed of a meta-logic and a proof system has been developed and implemented in a new interactive prover, the Squirrel prover [BDJ+20]. This prover takes as input protocols specified in the applied pi-calculus, and has already been used to perform a number of case studies.

**Objectives of the internship.** The main objective of this internship will be to contribute to the development of the Squirrel prover. This can be achieved in various ways depending on the skills and expectations of the student.

1. In this framework, protocols are modelled through a finite set of actions equipped with a dependency relation, which constrains the order of execution of actions. Each action represents a basic step of the protocol where the attacker provides an input, a condition is checked, and finally an output is emitted. For sake of usability, the Squirrel prover takes as input protocols specified in a more convenient language (close to the applied pi-calculus commonly used to specify protocols), and translates this applied pi-calculus specification into a protocol as a set of actions. This translation is already implemented but has never been proved correct. It would be interesting to address the formal semantics of these applied pi-calculus protocols, and the study of their translations to the internal representation as sets of actions.

2. Regarding case studies, it will be interesting to also conduct additional case studies. A rather simple objective could be to analyse classical protocols (Needham-Schroeder-Lowe, Wide Mouth Frog, Yahalom, etc) for which a security analysis has already been conducted in a framework based on similar concepts [Sce15]. This may require the development of new tactics (based on existing axioms) in the Squirrel prover.

3. Currently, the CCSA approach only provides asymptotic guarantees for each trace (with an asymptotic bound that may depend on the trace). This is due to the fact that the security parameter (on which any computational proof depends) is abtracted away in the CCSA approach. In order to make this approach more attractive to cryptographers, it would be interesting to find conditions under which a security proof in the CCSA model, which is for any number of sessions independent from the security parameter, can be lifted to a security proof for a polynomial number of sessions.

**Expected skills.** We are looking for candidates with good skills in Foundations of Computer Science (logic, automated deduction, concurrency theory…). Some knowledge in security is an asset but is not mandatory. The candidate will assimilate this knowledge during the internship.

This internship may also lead to a PhD thesis on similar topics.

## References

[BC14] Gergei Bana and Hubert Comon-Lundh. A computationally complete symbolic attacker for equivalence properties. In *ACM Conference on Computer and Communications Security*, pages 609–620. ACM, 2014.

[BDJ+20] David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutos, and Solène Moreau. An interactive prover for protocol verification in the computational model. Technical report, 2020.

[CK11] Véronique Cortier and Steve Kremer, editors. *Formal Models and Techniques for Analyzing Security Protocols*, volume 5 of *Cryptology and Information Security Series.* IOS Press, 2011.

[CK17] Hubert Comon and Adrien Koutsos. Formal computational unlinkability proofs of RFID protocols. In *CSF*, pages 100–114. IEEE Computer Society, 2017.

[Sce15] Guillaume Scerri. *Proofs of security protocols revisited. (Les preuves de protocoles cryptographiques revisitées)*. PhD thesis, École normale supérieure de Cachan, France, 2015.