Internship project
Composition in the Squirrel Prover

Charlie Jacomme

March 14, 2024

General Context The Squirrel prover [1] is an interactive prover designed to prove the security of cryptographic protocols, by only working inside the context of a logic and abstracting away the usual probabilistic and reductionist arguments. If the original foundations of Squirrel's logic were a first order logic designed by Bana and Comon [3], it was recently extended to a full high-order logic [2]. This recent extension opens up new possibility in terms of expressivity, that we aim to explore here.

A composition result In a previous work [4], we developed a framework for making compositional proofs in cryptographic proofs. To illustrate this result, consider a protocol P, for which we want to prove a security property ϕ_P, but in the context of any attacker $\mathcal{A}$. Informally, our goal is then to prove a formula of the form $\forall \mathcal{A}. P \| A \models \phi_P$, where $\mathcal{A}$ can run in parallel to P. In practice, we often have protocols that are running in parallel to one another, and we have to prove something like $\forall \mathcal{A}. P \| Q \| A \models \phi_P$ for some Q. First, notice that if Q can be fully simulated by $\mathcal{A}$ ($\mathcal{A}$ can emulate all the computations of Q itself), then one can push Q inside the attacker and ignore it. But, in practice, it is often the case that P and Q share some secret data sk, and the attacker cannot simulate the program Q. The core idea of our framework is to replace Q by a simpler piece of code, more generic, that will give partial access to sk to $\mathcal{A}$, enough so that $\mathcal{A}$ can simulate Q, but not too much so that P is still secure. We call this technique oracle simulation, where the attacker must be able to simulate Q when given access to some oracle O_{sk}.

By building on this core technique, we developed several proof techniques, for parallel and sequential composition, for many-to-one session security reductions, and for key-exchange applications.

The project The high order logic behind Squirrel now enables one to directly specify in the logic first-order functions, lambda terms, that behave as oracles given to the attacker. In this project, we aim to leverage this new feature to naturally re-cast the composition result inside Squirrel.

The first step will be to express the basic oracle simulation result and to provide a valid inference rule for the logic, that allows to replace pieces of the

protocol by simpler first-order term under some simulation condition. This first step will already require a basic understanding of the two main papers and demonstrate the student capability to assimilate new notions and work within their formalism.

Then, as a follow-up, we will explore whether the multiple developments in the framework can be formally proved and used directly inside Squirrel, with the goal of providing a fully integrated composition framework as a library of the tool.

We have already identified some of the challenges underlying this project, clearly making it a research project requiring the development and study of novel ideas:

- special care need to be taken when consider the simulation notion, as one may have to replace some random samplings of the protocol by random sampling of the attacker or some oracle. Correctly managing those dependencies is essential.
- Squirrel models protocol as set of actions, whose executions are described and reasoned over as terms. Lifting the preliminary simulation notions to actions and recursive term will yield new questions.

References

[1] David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutsos, and Solène Moreau. An Interactive Prover for Protocol Verification in the Computational Model. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 537–554, May 2021. ISSN: 2375-1207.

[2] David Baelde, Adrien Koutsos, and Joseph Lallemand. A Higher-Order Indistinguishability Logic for Cryptographic Reasoning. In *2023 38th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, June 2023.

[3] Gergei Bana and Hubert Comon-Lundh. A Computationally Complete Symbolic Attacker for Equivalence Properties. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 21st ACM Conference on Computer and Communications Security (CCS'14)*, pages 609–620, Scottsdale, Arizona, USA, November 2014. ACM Press.

[4] Hubert Comon, Charlie Jacomme, and Guillaume Scerri. Oracle Simulation: A Technique for Protocol Composition with Long Term Shared Secrets. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, CCS '20, pages 1427–1444, New York, NY, USA, October 2020. Association for Computing Machinery.