Safe Composition of Jasmin & Rust for High-Assurance Cryptography

Keywords Compilation, formal methods, high-assurance cryptography, static analysis
City and country Nancy, France
Team and research laboratory Team Pesto, at Loria lab (Inria Nancy, CNRS, Université de Lorraine)
Name and address of the advisor Vincent Laporte, Vincent.Laporte@inria.fr
Name and address of the head of the laboratory Yannick Toussaint, yannick.toussaint@loria.fr
Funding The internship is supported by the PÉPR "Cybersécurité" (France 2030 program managed by the French National Research Agency under grant agreement No. ANR-22-PECY-0006).

Context

Implementing security primitives (encryption, signature, etc.) poses a triple challenge: the implementation must be correct, run fast, and be itself secure, e.g., do not leak sensitive data even when executed in hostile environments. The Jasmin language (Almeida et al. 2017) aims at being well suited for this task: write high-performance low-level programs and enable their formal verification. For instance, the Jasmin infrastructure has been used to provide formal security guaranties of a very efficient implementation of the SHA3 hash function (Almeida et al. 2019). The Jasmin framework has also enabled formal security proofs of efficient implementations of the ML-KEM post-quantum primitive (Almeida et al. 2023, 2024).

The success of the language relies on the equilibrium between the high-level abstractions that ease the implementation of programs and reasoning about them, and the predictability of the compiler that gives control on performances to the programmer. The correctness proof of the compiler (mechanized in the Coq proof assistant) justifies the formal reasoning done at the source level.

Objective of the internship

Jasmin implementations provide core functionalities (e.g., computing an authentication tag, encrypting a message) that are meant to be used to build larger applications, like a messaging system or a web browser. Said applications are implemented in other programming languages (C, OCaml, Rust...). The goal of the internship is to study what security guaranties proved on a Jasmin program still apply in the context of a larger application.

The formal verification of a Jasmin program assumes that this program is executed in a safe context. When it is linked with an unknown application program, it is usually difficult (and possibly inefficient) to ensure that all executions of the Jasmin program are actually safe. However when the context is written in a strongly typed language, it is possible to rely on the type system to reason locally and ensure safety by typing.

Reciprocally, from the point-of-view of the application, it should also be safe to call the Jasmin program. Although said Jasmin program is unlikely to be well-typed in the application language, its overall execution should comply with its strong type-safety discipline. The challenge here is to ensure that a (safe) Jasmin program does not break any typing invariant of the application.

Finally, the RustBelt model (Jung et al. 2018) is a formal model of the Rust type system that has been used to verify programs that locally escape the strong typing discipline of Rust. Such a model can serve as a foundation to justify the soundness of a composition of Rust and Jasmin programs and to mechanize such a proof in the Coq proof assistant.

More precisely, the intern will:

- study how to ensure the safety of a Jasmin program when called from a statically typed programming language (namely Rust);
- study under which conditions a Jasmin program can be safely encapsulated under a Rust type;
- design and prototype a static analyzer to check that these conditions are met;
- study the RustBelt model of Rust and eventually justify the previous analysis in this framework.

Expected ability of the student

We expect mathematical maturity, basic knowledge in logic, basic theoretical computer science, some understanding of programming language semantics. Knowledge in security and cryptography is not mandatory. Ability to program in OCaml would be appreciated.

Note that there is funding available to extend this internship towards a PhD, would the candidate be interested.

Bibliographic references

Almeida, J. B., Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Vincent Laporte, Jean-Christophe Léchenet, Tiago Oliveira, et al. 2023. "Formally verifying Kyber: Episode IV: Implementation correctness." In *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2023:164–93. 3. Praha, Czech Republic: IACR. https://doi.org/10.46586/tches.v2023.i3.164-193.

Almeida, J. B., M. Barbosa, G. Barthe, A. Blot, B. Grégoire, V. Laporte, T. Oliveira, H. Pacheco, B. Schmidt, and P.-Y. Strub. 2017. "Jasmin: High-Assurance and High-Speed Cryptography." In *Proceedings of the 24th ACM Conference on Computer and Communications Security*, (*CCS*).

Almeida, J. B., C. Baritel-Ruet, M. Barbosa, G. Barthe, F. Dupressoir, B. Grégoire, V. Laporte, T. Oliveira, A. Stoughton, and P.-Y. Strub. 2019. "Machine-Checked Proofs for Cryptographic Standards: Indifferentiability of Sponge and Secure High-Assurance Implementations of SHA-3." In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, (*CCS*).

Almeida, J. B., Santiago Arranz Olmos, Manuel Barbosa, Gilles Barthe, François Dupressoir, Benjamin Grégoire, Vincent Laporte, et al. 2024. "Formally verifying Kyber: Episode V: Machine-checked IND-CCA security and correctness of ML-KEM in EasyCrypt." In *Crypto 2024*. Vol. 14921. IACR. https://doi.org/10.1007/978-3-031-68379-4_12.

Jung, R., J.-H. Jourdan, R. Krebbers, and D. Dreyer. 2018. "RustBelt: Securing the Foundations of the Rust Programming Language." *Proc. ACM Program. Lang.* 2 (POPL): 66:1–34. https://doi.org/10.1145/3158154.