Can Verification of Cryptographic Libraries be liberated from the von Neumann Style?

Marcel Fourné
Lehrstuhl Formale Methoden der Informatik
Universität Duisburg-Essen

Functional Programming has been used to generate highly efficient and highly verified implementations of operating systems (Klein *et al.*, 2009) and cryptographic implementations (Zinzindohoué *et al.*, 2017) by generating a subset of the C programming language and using various compilers to generate fast or safe code. More common verified C implementations can exhibit problems in practice due to that choice (Kaufmann *et al.*, 2016).
Branching behaviour is now shown to be more succeptible to leaking secret information from programs (Kocher *et al.*, 2019). This branching exhibits an observable side-effect of otherwise pure computations. The question comes to mind: Can we use purely functional programming to get rid of those side-effects? Even more so, will the security arguments against sidechannel feasibility be more rigorous than those of implementations which generate C code, or even those written purely in C?
My research is using the Haskell programming language as a testbed for this due to a strict type system which lends itself to secure programming practices (Peyton Jones, 2012). A new Ed25119 (Bernstein *et al.*, 2012) implementation with focus on avoiding branching behaviour on secrets is being developed and will be used as an example for verification of similar applications. This talk will introduce the problem setting surrounding my research question.

crypto day matters 29 (2018)
Beiträge vom 29. Krypto-Tag, 06.-07.09.2018, Bosch, Renningen.
doi:10.18420/cdm-2018-29-26

References

Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe & Bo-Yin Yang (2012). High-speed high-security signatures. *Journal of Cryptographic Engineering* **2**(2), 77–89. ISSN 2190-8516. URL `https://doi.org/10.1007/s13389-012-0027-1`.

Thierry Kaufmann, Hervé Pelletier, Serge Vaudenay & Karine Villegas (2016). When constant-time source yields variable-time binary: Exploiting curve25519-donna built with msvc 2015. In *International Conference on Cryptology and Network Security*, 573–582. Springer.

Gerwin Klein, Kevin Elphinstone, Gernot Heiser, June Andronick, David Cock, Philip Derrin, Dhammika Elkaduwe, Kai Engelhardt, Rafal Kolanski, Michael Norrish, Thomas Sewell, Harvey Tuch & Simon Winwood (2009). seL4: Formal Verification of an OS Kernel.

In *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles*, SOSP '09, 207–220. ACM, New York, NY, USA. ISBN 978-1-60558-752-3. URL `http://doi.acm.org/10.1145/1629575.1629596`.

Paul Kocher, Jann Horn, Anders Fogh, , Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz & Yuval Yarom (2019). Spectre Attacks: Exploiting Speculative Execution. In *40th IEEE Symposium on Security and Privacy (S&P'19)*.

Simon Peyton Jones (2012). Safe Haskell. In *Haskell '12: Proceedings of the Fifth ACM SIGPLAN Symposium on Haskell*. ACM. URL `https://www.microsoft.com/en-us/research/publication/safe-haskell/`.

Jean-Karim Zinzindohoué, Karthikeyan Bhargavan, Jonathan Protzenko & Benjamin Beurdouche (2017). HACL*: A verified modern cryptographic library. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1789–1806. ACM.