Formalisation and proofs about Kuifje
A Quantitative Information Flow aware language

Jack Drury

University of New South Wales, Australia

Information Flow is the technical name for (unintended) security lapses of programs or other computer-based devices: informally they are called "leaks". A recent trend has been to investigate not whether information escapes from a program, but rather whether there could be a leak from a proposed implementation in circumstances where the specification would not leak. And an even more up-to-date approach is to determine from the source code how much information would leak, and then to ask whether a proposed implementation might leak more than its specification. This is Quantitative Information Flow, or QIF.

Kuifje is a Quantitative Information Flow aware programming language with a compiler written in Haskell and meant to allow for reasoning about these security properties [1, 2].

In order to trust the outcomes of the reasoning performed by Kuifje, it is important to prove that it's semantics maintain certain algebraic properties. One way to do this would be to formalise Kuifje in an interactive theorem prover (such as Isabelle/HOL [3]) and then use the theorem prover to prove these properties.

The aim of this project is to formalize Kuifje in Isabelle/HOL and prove that the language semantics maintains certain algebraic properties. If time allows, we plan to use these theorems to verify that some Kuifje programs that are meant to be secure are indeed secure.

References

1. Morgan, C., Gibbons, J., Mciver, A., Schrijvers, T.: Quantitative information flow with monads in haskell. In: Foundations of Probabilistic Programming. CUP (2019, to appear)
2. M. Bognar: Kuifje: a Quantitative Information Flow aware programming language, http://hackage.haskell.org/package/kuifje (2019)
3. T. Nipkow, L.C. Paulson & M. Wenzel: Isabelle/HOL — A Proof Assistant for Higher-Order Logic. LNCS 2283, Springer, doi:10.1007/3-540-45949-9. (2002)