Certified Systems Development with Ynot

Ryan Wisnesky Gregory Malecha

March 18, 2010

In this talk we describe our experience building a provably correct (certified) web-based course gradebook application in Ynot, an axiomatic extension to the Coq proof assistant. We demonstrate that Ynot can be used to implement certified systems in a way similar to writing ML or Haskell code, including use of effectful, imperative features such as pointers, files, and socket I/O. The proof of system correctness is developed interactively with the programmer, imposes no runtime overhead, and can be verified in minutes by a several hundred-line typechecker.

Ynot can be thought of as dependently typed Haskell with an IO monad that is indexed by pre and postcondtions in the style of Hoare logic. Memory effects are reasoned about using separation logic, and we extend Ynot by adding support for reasoning about externally observable events like network and file I/O using a trace semantics. We demonstrate how certification of high level properties like privacy guarantees can be effectively isolated from certification of low level properties like memory safety and parsing correctness by leveraging higher-order functions and Coq's proof-search language.