Deep Specifications and Certified Abstraction Layers

Submitted by Ronghui Gu on Thu, 04/23/2015 - 6:36pm. Contributor: Ronghui Gu

Abstract:
Modern computer systems consist of a multitude of abstraction layers (e.g., OS kernels, hypervisors, device drivers, network protocols), each of which defines an interface that hides the implementation details of a particular set of functionality. Client programs built on top of each layer can be understood solely based on the interface, independent of the layer implementation. Despite their obvious importance, abstraction layers have mostly been treated as a system concept; they have almost never been formally specified or verified. This makes it difficult to establish strong correctness properties, and to scale program verification across multiple layers.

In this talk, I will present a novel language-based account of abstraction layers and show that they correspond to a strong form of abstraction over a particularly rich class of specifications which we call deep specifications. Just as data abstraction in typed functional languages leads to the important representation independence property, abstraction over deep specification is characterized by an important implementation independence property: any two implementations of the same deep specification must have contextually equivalent behaviors. We present a new layer calculus showing how to formally specify, program, verify, and compose abstraction layers. We show how to instantiate the layer calculus in realistic programming languages such as C and assembly, and how to adapt the CompCert verified compiler to compile certified C layers such that they can be linked with assembly layers. Using these new languages and tools, we have successfully developed multiple certified OS kernels in the Coq proof assistant, the most realistic of which consists of 37 abstraction layers, took less than one person year to develop, and can boot a version of Linux as a guest.

This talk represents joint work with Jeremie Koenig, Tahina Ramananandro, Zhong Shao, Newman Wu, Shu-Chun Weng, Haozhong Zhang, and Yu Guo. A paper documenting this result was published in POPL 2015. Both the paper, the technical report, and the presentation slides can be found at the URL: <http://flint.cs.yale.edu/publications/dscal.html>

Biography:
Ronghui Gu is a forth-year Ph.D. student at Yale University and is supervised by professor Zhong Shao. He received his Bachelor degree from Tsinghua University in 2011. Ronghui is interested in building reliable and secure computer software, especially the trusted low level systems. Together with his colleagues at Yale FLINT group, he worked on a general framework, named CertiKOS, to verify low level systems by formally dividing a complex system into a stack of abstraction layers. Using the CertiKOS framework, they verified four versions of practical OS kernels, hypervisors within 1.5 person years.

License: Creative Commons 2.5

Other available formats: