Reference monitors for proof-carrying code

Simon Winwood
sjw@cse.unsw.edu.au

Proof carrying code (PCC) is inherently trustworthy, independent of its origin or previous opportunities for tampering. The guarantees provided by PCC are, however, not universal: they are relative to a *security policy* agreed upon by the code producer and consumer. It is the code producer's obligation to annotate the code with a *proof object* that establishes the code's compliance with the security policy. This proof object, consisting of steps in a *formal logic,* can be checked with a simple proof checker. Thus, the trustworthiness of the code can be established with mathematical rigour.

Existing research into the generation of proof-carrying code focuses on security policies which can be derived from properties of high-level languages and their type systems, such as type safety, control and memory safety, and space/time guarantees. In this talk we discuss the extension of PCC systems to more general security policies through the use of reference monitors.

These monitors are synthesised from policies expressed in the safety fragment of *linear temporal logic* (LTL). Thus policies can refer to past events, allowing the enforcement of a number of interesting security policies. In contrast to previous work on generating reference monitors from temporal logics, we simultaneously generate a proof object that demonstrates that the generated code enforces the security policy. Such proofs allow the reference monitors to be used in PCC systems.

This talk will focus on the automatic generation of these proofs, along with a framework for expressing and checking monitors and proofs.