From Formal Specification to an Implementation of an Embedded Verifier for Java Card

Ludovic Casset
Gemplus Research Lab
BP 100 Gémenos Cedex France

+33(0)4 42 36 40 98
+33(0)4 42 36 55 55
ludovic.casset@gemplus.com

The Matisse[1] project aims to propose methodologies, tools and techniques for using formal methods in an industrial concerns. Seven partners work together on three different case studies related to transport, health care and smart card. The presentation will focus on the smart card case study which concerns the formal specification and the implementation of the Java Card verifier.
The Java security policy is implemented by security components such as the Java Virtual Machine (JVM), the API, the verifier, the loader. It is of prime importance to ensure that the implementation of these components is in accordance with their specifications. In the smart card case study of the Matisse project, one aims to model the Java Card verifier taking into account the whole language and using the B method. The verifier helps the smart card ensuring its own security by checking the downloaded applet before its on-card installation: if the verification fails, the applet is not installed into the card. The byte code verification aims to enforce static constraints on downloaded byte code. Those constraints ensure that the byte code can be safely executed by the virtual machine, and cannot bypass the higher-level security mechanisms. The byte code verification consists in a static analysis of the downloaded applet and ensures that the downloaded applet file is a valid file, there are no stack overflow or underflow, the execution flow is confined to valid byte code, each instruction argument is of the correct type and methods calls are performed in accordance with their visibility attributes (`public`, `protected`, etc…). The first point corresponds to the structural verification, and the next points are performed by the type verification. One provides a formal model of each verifier, i.e. the structural verifier and the type verifier.
The type verifier is entirely modelled in B. Its development is made simpler as it relies on many services provided by the structural verifier and expressed through an interface. One proves that its implementation is consistent with its specification. The B method allows us to provide a very abstract specification that is split in several modules. In fact, we do not use a simple scheme where the specification is in the abstract machine which is refined and implemented. We provide a formal specification which is made of several modules (abstract machines, refinements and implementation). This formal specification is then refined in order to obtain an implementation.
The structural verifier is not entirely developed in B but relies on basic blocks such as the memory management, and file representation within the smart card. The structural verifier implements the interface of services produced by the type verifier. It also includes internal and external tests. Internal tests correspond to tests related to each component of a CAP file, i.e. Applet, Class, etc… We refer to tests between components as external tests. External tests correspond to interdependencies between components, like shared information or references from a component to another. All the tests aim to ensure the correctness of the CAP file and consistency of data contained in the CAP file.
Both verifiers are then translated into C code automatically from B specification. The C code is compiled and embedded into an ATMEL chip and a demonstration is available. We then discuss the result of such an experiment. In particular, the architecture needed to perform this development and the proof effort is analysed. Moreover, we discuss the integration of formal development into a smart card and with pre-existing traditional developments.

[1] European IST Project MATISSE number *IST-1999-11435*.