D6.2 Tool support for the Programming model

(executive summary)

Frank Piessens, Bart Jacobs, Jan Smans, Pieter Philippaerts (K.U.Leuven) Fabio Massacci, Michela Angeli (UNITN)

Document information

Document Number	D6.2
Document Title	Tool support for the programming model
Version	2.1
Status	Final
Work Package	WP 6
Deliverable Type	Prototype
Contractual Date of Delivery	31 January 2011
Actual Date of Delivery	31 January 2011
Responsible Unit	KUL
Contributors	
Keyword List	Programming models, verification
Dissemination level	PU

Document change record

Version	Date	Status	Author (Unit)	Description
1.0	30 Nov 2010	Draft	Frank Piessens, Bart Jacobs, Jan Smans, Pieter Philippaerts (KUL)	
2.0	23 Dec 2010	Release Candidate	Frank Piessens (KUL), Fabio Massacci (UNITN)	Processed comments from Fabio on first draft
2.1	13 Jan 2011	Final	Frank Piessens (KUL), Michela Angeli (UNITN)	Quality check completed – minor remarks

Executive summary

This document summarizes the work performed in Task 6.2 of Work Package 6 of the SecureChange project funded by the European Commission within the Seventh Framework Programme.

The overall objective of Work Package 6 is the development of verification techniques for evolving systems, with a strong focus on the development time and run time phases of the software lifecycle. Tasks 6.1 and 6.2 focus on development time. The objective of Task 6.1 was the development of techniques that can ensure the absence of classes of vulnerabilities using formal verification. Deliverable 6.1, delivered after the first year, developed the theory of how to extend a separation-logic based verifier so that it can verify soundly absence of several classes of bugs, even in the presence of unchecked exceptions and dynamic code loading and unloading.

The main objective of Task 6.2 is the implementation of a prototype of such a verifier, and this deliverable D6.2 (a tool deliverable) delivers:

- A binary distribution of this prototype verifier (called VeriFast)
- A tool paper describing the tool at a high level
- Some sample code from the POPS case study that has been successfully verified

In order to support validation of the verifier on the HOMES and POPS case studies, we have implemented both Java and C front ends for the verifier, and we provide a full implementation of the techniques for dynamic code loading and unloading developed in Task 6.1.

The tool is sufficiently mature that it can verify real Java Card code taken from the POPS case study. Validation of the prototype on the POPS case study has started. For this case study, one of the concerns is robustness (absence of denial-of-service issues) when software updates happen on the card. The prototype tool is being used to verify absence of runtime exceptions and infinite loops in Java Card applets that are loaded on a multi-application smartcard. Several applets have already been verified. A preliminary evaluation shows that verification is efficient, but that annotation overhead is high. In the third year of the project, effort will be spent to reduce this annotation overhead.

Validation of the tool on the HOMES case study is planned for the third year. For the HOMES case study, the tool will verify the *secure extensibility* property for core security module updates of the home gateway. The subset of C that is supported by the prototype tool should be further extended before this validation can start. But initial experiments with simplified models of the code to be verified are promising : they show that the verifier can verify the safety of C code that loads and unloads modules efficiently.