Security-by-Contract: Toward a Semantics for Digital Signatures on Mobile Code*

N. Dragoni, F. Massacci, K. Naliuka, and I. Siahaan

Department of Information and Communication Technologies
University of Trento, Italy
surname@dit.unitn.it

Abstract. In this paper we propose the notion of *security-by-contract*, a mobile contract that an application carries with itself. The key idea of the framework is that a digital signature should not just certify the origin of the code but rather bind together the code with a contract. We provide a description of the overall life-cycle of mobile code in the setting of security-by-contract, describe a tentative structure for a contractual language and propose a number of algorithms for one of the key steps in the process, the *contract-policy matching* issue. We argue that security-by-contract would provide a semantics for digital signatures on mobile code thus being a step in the transition from trusted code to trustworthy code.

1 Introduction

Mobile devices are increasingly popular and powerful. Yet, the growth in computing power of nomadic devices has not been supported by a comparable growth in available software: on high-end mobile phones we cannot even remotely find the amount of third party software that was available on our old PC.

One of the reasons for this lack of applications is also the security model adopted for mobile phones. The current security model is exemplified by the JAVA MIDP 2.0 approach and is based on *trust relationships*: mobile code is run if its origin is trusted. This essentially boils down to *mobile code is accepted if it is digitally signed by a trusted party*. The level of trust of the "trusted party" determines the privileges of the code by essentially segregating it into appropriate trust domain.

The problem with trust relationship, i.e. digital signatures on mobile code, is twofold. At first we can only reject or accept the signature. This means that inter-operability in a domain is either total or not existing: an application from a not-so-trusted source can be denied network access, but it cannot be denied access to a specific protocol, or to a specific domain. E.g. if a payment service is available on the platform and an application for paying parking meters is loaded, the user cannot block the application from performing large payments.

The second (and major) problem, is that *there is no semantics attached to the signature*. This is a problem for both code producers and consumers.

From the point of view of mobile code consumers they must essentially accept the code "as-is" without the possibility of making informed decisions. One might well trust

* Research partly supported by the project EU-IST-STREP-S3MS.

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 297–312, 2007.

SuperGame Inc. to provide excellent games and yet might decide to rule out games that keep playing while the battery fells below 20%. At present such choice is not possible.

From the point of view of the code producer they produce code with unbounded liability. They cannot declare which security actions the code will do, by signing the code they essentially declare that they did it. The consequence is that injecting an application in the mobile market is a time consuming operation as SME developers must essentially convince the operators that their code will not do anything harmful.

1.1 Contribution of the Paper

We propose in this paper the notion of *security-by-contract* (as in programming-by contract [1]): the digital signature should not just certify the origin of the code but rather bind together the code with a contract. Loosely speaking, a *contract* contains a description of the relevant features of the application and the relevant interactions with its host platform. A mobile platform could specify platform contractual requirements, a *policy*[1], which should be matched by the application's contract. Among the relevant features, one can list fine-grained resource control (e.g. silently initiate a phone call or send a SMS), memory usage, secure and insecure web connections, user privacy protection, confidentiality of application data, constraints on access from other applications already on the platform.

We provide here a description of the overall life-cycle of mobile code in the setting of security-by-contract, describe a tentative structure for a contractual language and propose a number of algorithms for one of the key steps in the process, namely the issue of *contract-policy matching*.

We argue that security-by-contract would provide a semantics for digital signatures on mobile code thus being a step in the transition from trusted code to trustworthy code.

The research is performed within the limits of the European project "Security of Software and Services for Mobile Systems" (S3MS)[2].

The rest of the paper is organized as follows. In Section 2 we present the security-by-contract framework providing a description of the overall life-cycle of mobile code in this setting. In Section 3 we describe typical security requirements to mobile applications. In Section 4 we focus on contract specification defining also the notion of contract-policy matching. Then in Section 5 we propose an algorithm for contract-policy matching based on the contractual language of Section 4. We end the paper discussing related work and conclusions.

2 The Security-by-Contract Life-Cycle

The framework of security-by-contract for mobile code is essentially shaped by three groups of stake-holders: mobile operator, service provider and/or developer, mobile user. This is shown in Fig. 1.

The mobile code developers are responsible to provide a description of the security behavior that their code provides.

[1] In the sequel we will refer to policy as the security requirements on the platform side and by contract the security claims made by the mobile code.

[2] More information concerning this project can be acquired at http:\\www.s3ms.org.