Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 331 (2017) 71–85

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Software Watermarking: A Semantics-based Approach

Mila Dalla Preda[1,2,4] Michele Pasqua[1,3]

Department of Computer Science
University of Verona
Strada le Grazie 15, 37134, Verona, ITALY

Abstract

Software watermarking is a defence technique used to prevent software piracy by embedding a signature, i.e., an identifier reliably representing the owner, in the code. When an illegal copy is made, the ownership can be claimed by extracting this identifier. The signature has to be hidden inside the program and it has to be difficult for an attacker to detect, tamper or remove it. In this paper we show how the ability of the attacker to identify the signature can be modelled in the framework of abstract interpretation as a completeness property. We view attackers as abstract interpreters that can precisely observe only the properties for which they are complete. In this setting, hiding a signature in the code corresponds to inserting it in terms of a semantic property that can be retrieved only by attackers that are complete for it. Indeed, any abstract interpreter that is not complete for the property specifying the signature cannot detect, tamper or remove it. The goal of this work is to introduce a formal framework for the modelling, at a semantic level, of software watermarking techniques and their quality features.

Keywords: Software Watermarking, Abstract Interpretation, Program semantics, Program transformation

1 Introduction

Software developers are interested in protecting the intellectual property of their products against software piracy, namely to prevent the illegal reuse of their code. Software watermarking is a technique for embedding a signature, i.e., an identifier reliably representing the owner, in a cover program. This allows software developers to prove their ownership by extracting their signature from the pirated copies. In the last two decades researchers have developed a variety of software watermarking techniques (e.g., [3,4]) that can be classified in three main categories according to their extraction process: static, dynamic and abstract watermarking. *Static*

[1] We would like to thank Roberto Giacobazzi for the initial discussions on this work and Isabella Mastroeni for the discussions on high-order abstract non-interference.

[2] Email: mila.dallapreda@univr.it

[3] Email: michele.pasqua@univr.it

[4] This work has been supported by the MIUR FIRB 2013 project FACE RBFR13AJFT.

http://dx.doi.org/10.1016/j.entcs.2017.02.005

1571-0661/© 2017 The Author(s). Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

watermarking inserts signatures in the cover program either as data or code and then extracts them statically, namely without executing the code [4]. Conversely, *dynamic watermarking* inserts signatures in the program execution state (i.e., in its semantics) and the extraction process requires the execution of the program, often on a special enabling input [4]. *Abstract watermarking*, introduced in [10], encodes the signature in such a way that it could be extracted only by a suitable abstract execution of the program. A watermarking scheme is typically evaluated w.r.t. the following features: credibility (how strongly it proves authorship), secrecy (how difficult it is to extract the mark), transparence (how difficult it is to realize that a program is marked), accuracy (observational equivalence of the marked and original program), resilience to attacks (how difficult it is to compromise the correct extraction of the signature) and data-rate (amount of information that can be encoded). The quality of each existing watermarking technique is specified in terms of their features that are typically claimed to hold w.r.t. the peculiar embedding and extraction methods. There exists a variety of embedding and extraction algorithms that often work on different objects (control flow graph, variables, registers, etc.) and this makes it difficult to compare the efficiency of different watermarking systems. It is therefore difficult to formally prove and compare limits and potentialities of the different watermarking systems and to decide which one is better to use in a given scenario.

These problems derive also by the fact that, at the state of the art, there is a poor theoretical investigation about software watermarking. The concept was formally defined in [1] and, in the same work, the authors showed that the existence of indistinguishability obfuscators implies that software watermarking cannot exist. Furthermore the recent candidate construction of an indistinguishability obfuscator [12] lowers the hope of building meaningful watermarking scheme. Fortunately this impossibility result relies on the fact that the signed program computes the same function as the original program. Indeed, in [1] the authors suggested that if we relax this last constraint, i.e., we require that the watermarking process has only to preserve an "approximation" of original program's functionality, then positive results may be possible. This naturally leads to reason about software watermarking at semantic level, as we do in the present work.

A first attempt to provide a formal definition, in the semantics setting, of a watermarking system has been proposed in [13]. Here the author introduced the idea that static and dynamic watermarking are instances of abstract watermarking. Intuitively, the latter can be seen like static watermarking because the extraction of the signature requires no execution. But, at the same time, it can be seen like dynamic watermarking because the signature is hidden in the semantics. So all these three types of techniques could be seen as particular instances of a common watermarking scheme based on program semantics and abstract interpretation.

In this work, we start from that intuition and we transform the scheme proposed in [13] in a formal and consistent definition of what a software watermarking system is. The idea is to model the embedding of the secret signature s as the encoding of s in a semantic property $\overline{\mathfrak{M}}(s)$ that is then inserted in the semantics of the cover program. In this setting, the extraction process requires an analysis of the marked code

Download English Version:

https://daneshyari.com/en/article/4950027

Download Persian Version:

https://daneshyari.com/article/4950027

Daneshyari.com