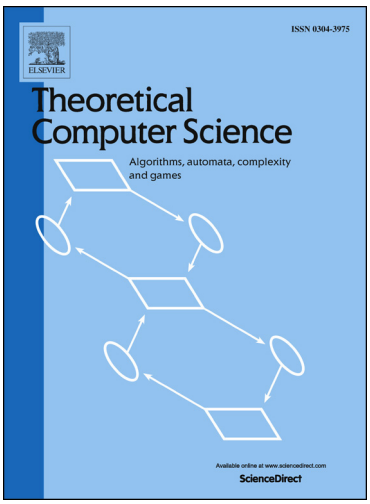
Accepted Manuscript



On detecting environment sensitivity using slicing

Xiang Fu

PII: S0304-3975(16)30468-6
DOI: http://dx.doi.org/10.1016/j.tcs.2016.09.004
Reference: TCS 10918

To appear in: *Theoretical Computer Science*

Received date: 28 March 2016
Revised date: 5 September 2016
Accepted date: 8 September 2016

Please cite this article in press as: X. Fu, On detecting environment sensitivity using slicing, *Theoret. Comput. Sci.* (2016), http://dx.doi.org/10.1016/j.tcs.2016.09.004

On Detecting Environment Sensitivity Using Slicing

Xiang Fu

Xiang.Fu@hofstra.edu
Department of Computer Science
Hofstra University, Hempstead, NY 11549, USA

Abstract

Modern malware often evades debuggers, virtual machines, and emulators. It is interesting if one can observe their behavior difference using controlled environments. This paper formalizes the notion of environment sensitivity, and proposes two alternative semantics: one based on program trace and the other based on code coverage. Then it tackles the following question: can one minimize an environment sensitive program? The work presents progressive executable slice, a subprogram generated from a partial control but full data dependency closure of a program under study. It shows that a progressive slice can retain trace based environment sensitivity but not the code coverage sensitivity, for which, a special condition is needed for restraining the slice. The saturated trace set of a program is used as a cost indicator for observing its behavior difference. The paper shows that a progressive slice does not necessarily have a lower observation cost than its container program. To address it, a consistency condition is proposed. The paper introduces a reference algorithm for generating progressive slices, and discusses its approximation in practice.

Keywords: environment sensitivity, program slicing, malware analysis

1. Introduction

Modern malware analysis and intrusion detection systems (IDS) [1, 2, 3, 4, 5] often leverage virtual machines (VM), emulators, and debuggers for confined observation of malware. During this arms race, anti-analysis techniques are widely used in malware. As reported in [6], computer worms such as Agobot and Zlob