Science of Computer Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

A theory of integrating tamper evidence with stabilization

Reza Hajisheykhi [a,*], Ali Ebnenasir [b], Sandeep S. Kulkarni [a]

[a] Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA
[b] Department of Computer Science, Michigan Technological University, Houghton, MI 49931, USA

ARTICLE INFO

Article history:
Received 16 February 2016
Received in revised form 6 December 2016
Accepted 6 March 2017
Available online xxxx

Keywords:
Self-stabilization
Reactive systems
Adversary
Formal methods

ABSTRACT

We propose the notions of tamper-evident stabilization and flexible tamper-evident stabilization – that combine stabilization with the concept of tamper evidence – for computing systems. On the first glance, these notions are contradictory; stabilization requires that eventually the system functionality is fully restored whereas tamper evidence requires that the system functionality is permanently degraded in the event of tampering. Tamper-evident stabilization and flexible tamper-evident stabilization capture the intuition that the system will tolerate perturbations upto a limit. In the event that it is perturbed beyond that limit, it will exhibit permanent evidence of tampering, where it may provide reduced (possibly none) functionality. We compare tamper-evident stabilization with (conventional) stabilization and with active stabilization and propose two approaches to verify tamper-evident and flexible tamper-evident stabilizing programs in polynomial time in the size of state space. We demonstrate tamper-evident stabilization with two examples and point out some of its potential applications. We also demonstrate how approaches for designing stabilization can be used to design tamper-evident and flexible tamper-evident stabilizations. Finally, we study issues of composition in tamper-evident and flexible tamper-evident stabilizations and discuss how tamper-evident stabilization can effectively be used to provide tradeoff between fault-prevention and fault tolerance.

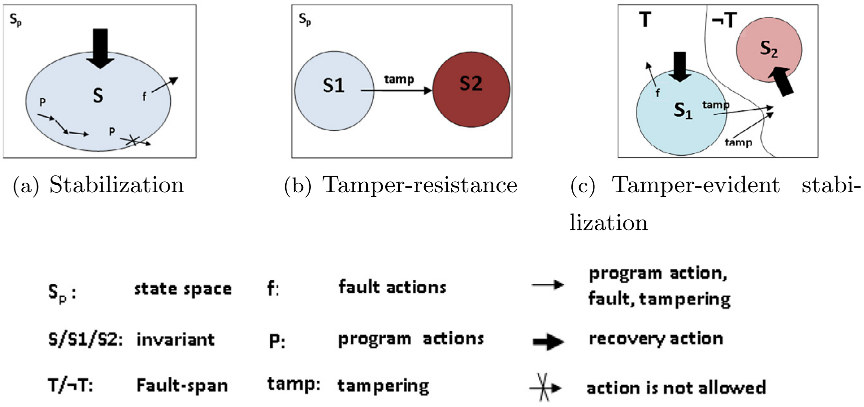
1. Introduction

A *stabilizing* system [15] ensures that it will recover to a set of legitimate states (S in Fig. 1(a), a.k.a invariant) even if it starts executing from an arbitrary state. For this reason, stabilization is often utilized as a type of fault-tolerance that requires recovery from unexpected transient faults. In other words, if the faults perturb the system to an arbitrary state, the goal of stabilizing systems is to ensure that the system will recover to the legitimate states with the assumption that no additional faults will occur. Nevertheless, stabilizing systems may not recover to legitimate states in the presence of tampering. Tamper-resistant systems are mostly utilized for secure chip designs (e.g. [37]). While *tamper-resistant* systems can prevent tampering to some degree, if tampering cannot be prevented, the system reaches a set of states where in the system is less functional/inoperable ($S2$ in Fig. 1(b)).

The notion of tamper resistance is contradictory to the notion of stabilization in that the notion of stabilization requires that in spite of any possible tampering the system inherently acquires its usefulness eventually. Therefore, in this paper, we combine these two seemingly conflicting concepts to benefit from the advantages of both. Specifically, we introduce the

[*] Corresponding author.
E-mail addresses: hajishey@cse.msu.edu (R. Hajisheykhi), aebnenas@mtu.edu (A. Ebnenasir), sandeep@cse.msu.edu (S.S. Kulkarni).

http://dx.doi.org/10.1016/j.scico.2017.03.001
0167-6423/© 2017 Elsevier B.V. All rights reserved.



(a) Stabilization (b) Tamper-resistance (c) Tamper-evident stabilization

Fig. 1. Stabilization vs. tamper-resistance.

notions of tamper-evident stabilizing and flexible tamper-evident stabilizing systems, and identify their properties in terms of composition, verification, and synthesis. The notions of tamper-evident and flexible tamper-evident stabilizing systems are motivated by the need for tamper-resistant systems that also stabilize. A tamper-resistant system ensures that an effort to tamper with the system makes the system less useful/inoperable (e.g., by zeroing out sensitive data in a chip or voiding the warranty).

Intuitively, the notions of tamper-evident and flexible tamper-evident stabilizations are based on the observation that all tamper-resistant systems tolerate some level of tampering without making the system less useful/inoperable. For example, a tamper-resistant chip may have a circuitry that does some rudimentary checks on the input and discards the input if the check fails. A communication protocol may use CRC to ensure that most random bit-flips in the message are tolerated without affecting the system. However, if the tampering is beyond some acceptable level then they become less useful/inoperable (see Fig. 1(b)). Based on this intuition, we observe that tamper-evident and flexible tamper-evident stabilizing systems will recover to their legitimate states if their perturbation is within an acceptable limit. However, if they are perturbed outside this boundary, they will make themselves inoperable. Moreover, when the systems enter the mode of making themselves inoperable, it is necessary that it cannot be prevented.

Thus, if a tamper-evident stabilizing system is outside its normal legitimate states, it is in one of two modes: *recovery mode*, where it is trying to restore itself to a legitimate state (the T area in Fig. 1(c)), or *tamper-evident mode*, where it is trying to make itself inoperable (the $\neg T$ area in Fig. 1(c)). The recovery mode is similar to the typical stabilizing systems in that the recovery should be guaranteed after external perturbations stop. However, in the tamper-evident mode, it is essential that the system makes itself inoperable even if outside perturbations continue.

To realize the last requirement, we need to make certain assumptions about what external perturbations can be performed during tamper-evident mode. For example, if these perturbations could restore the system to a legitimate state then designing tamper-evident stabilizing systems would be impossible. Hence, we view the system execution to consist of (1) program executions (in the absence of fault and adversary); (2) program executions in the presence of faults; and (3) program execution in the presence of *adversary*.

Faults are random events that perturb the system randomly and rarely. By contrast, the adversary is *actively* preventing the system from making itself inoperable. However, unlike faults, the adversary may not be able to perturb the system to an arbitrary state. Also, unlike faults, adversary may continue to execute forever. Even if the adversary executes forever, it is necessary that system actions have some fairness during execution. Hence, we assume that the system can make some number of steps between two steps of the adversary (in our formal definitions, we have this number of steps as strictly greater than 1).

Moreover, the notion of flexible tamper-evident stabilization is a more general definition of tamper-evident stabilization. Specifically, if a flexible tamper-evident stabilizing system is outside its legitimate states, it is in one of the recovery, tamper-evident, or *boundary* modes. The first two modes are similar to those in tamper-evident stabilizing systems. In the boundary mode, while the system can recover to either the set of legitimate states or tamper-evident states, an authorized operator decides to which set of states recovery should be achieved (depending on other environmental knowledge unavailable to the system).

The contributions of this paper are as follows. We

- formally define the notions of tamper-evident stabilization and flexible tamper-evident stabilization;
- compare the notion of tamper-evident stabilization with (conventional) stabilization and active stabilization, where a system stabilizes in spite of the interference of an adversary [12];
- present algorithms for the verification of tamper-evident stabilization and flexible tamper-evident stabilization;
- identify potential applications of tamper-evident stabilization and illustrate it with three examples;
- present some theorems about composing tamper-evident stabilizing and flexible tamper-evident stabilizing systems, and