Computer Networks 57 (2013) 2992–3008

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Using passive testing based on symbolic execution and slicing techniques: Application to the validation of communication protocols

Pramila Mouttappa [1], Stephane Maag *, Ana Cavalli *

Institut Mines-Telecom/TELECOM SudParis, CNRS UMR 5157, 9 rue Charles Fourier, F-91011 Evry Cedex, France

ARTICLE INFO

Article history:
Received 10 January 2013
Received in revised form 11 May 2013
Accepted 24 June 2013
Available online 11 July 2013

Keywords:
Passive testing
IOSTS
Parametric trace slicing
IMS/SIP architecture

ABSTRACT

This paper presents a new approach to perform passive testing based on the analysis of the control and data part of the system under test. Passive testing techniques are based on the observation and verification of properties on the behaviour of a system without interfering with its normal operation. Many passive testing techniques consider only the control part of the system and neglect data, or are confronted with an overwhelming amount of data values to process. In our approach, we consider control and data parts by integrating the concepts of symbolic execution and we improve trace analysis by introducing trace slicing techniques. Properties are described using Input–Output Symbolic Transition Systems (IOSTSs) and we illustrate in the paper how they can be tested on real execution traces optimizing the trace analysis. These properties can be designed to test the functional conformance of a protocol as well as security properties.

In addition to the theoretical approach, we have developed a software tool that implements the algorithms presented in this paper. Finally, as a proof of concept of our approach and tool we have applied the techniques to a real-life case study: the SIP protocol. In particular, the proposed techniques are applied to a set of real execution traces extracted from an IMS/SIP architecture.

1. Introduction

The advent of high-performance networks has led to the development of a new set of technologies and new communication protocols and services. Although the application of these protocols and services in real-time may be satisfiable, still there can be flaws in the implementation which could be exploited by an attacker to compromise the network. Testing is then an activity in which the testers try to guarantee that the protocol or service processes without fault or at least meets the requirements, but also to check security properties. Most of the formal testing [1–5] approaches consist in the generation of test cases that are applied to the implementation in order to check its correctness with respect to a specification. But, this is not always possible for large systems that operate continuously and where direct interfaces are not provided. Indeed, interfering with such systems can result in misbehavior of the system. In that case, passive testing is performed.

The usual approach of passive testing consists of recording the trace (i.e., sequence of exchange of messages) produced by the implementation under test and mapped to the property to be tested or specification if it exists. Passive testing helps to observe abnormal behavior in the implementation under test on the basis of observing any deviation from the predefined behavior. This deviation can also sometimes match with certain attack patterns. Moreover,

* Corresponding authors. Tel.: +33 618382082 (P. Mouttappa).
E-mail addresses: Pramila.Mouttappa@it-sudparis.eu (P. Mouttappa), Stephane.Maag@it-sudparis.eu (S. Maag), Ana.Cavalli@it-sudparis.eu (A. Cavalli).
[1] Principal author.

1389-1286/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2013.06.019

it is usually considered that the implementation is taken without knowledge of its internal state that is to say that we do not consider the event trace record to start from the initial state or a predefined state. In network protocols, communication is established by exchanging messages between the entities, where each entity can independently act as an emitter or receiver. As the systems evolve, messages become richer with data values. These messages are defined as control and data portions based on the function of the protocol. Many works on passive testing [6–9] are focused only on checking the control portion of the protocol without taking into account the data part. However, it may result in producing false positive verdicts as illustrated below with an example.

Let us consider a property, where a user *A* sends a *request* say, *Request (from:aron@ti.com, to:ben@info.com)* and expects for an *acknowledgement* response from user *B*, *Ack (from:aron@ti.com, to:ben@info.com)*. In Fig. 1a, the control portions are alone monitored. We observe from the system trace that the property of having a request followed by an acknowledgement is satisfied, hence the verdict for the trace results *pass*. But if we include the data portions of the messages then the verdict for the trace in Fig. 1b results *fail* or *inconclusive*. It fails because there is no acknowledgement response from user *B* as required by the property where we could only see the acknowledgement response from a different user, *Ack (from:carl@pouf.com, to:ben@info.com)* and the verdict is *inconclusive* if the length of the trace is not sufficient to prove the invariant. Hence the data relationship between messages must be given importance to avoid such false positive verdicts.

In order to overcome the above problem, the data part of the protocols must also be taken into account. This led to the development of specifications as extended finite state machines (EFSMs). In an EFSM, the transition can be expressed by an "if condition". If the trigger conditions are satisfied then the transition from one state to another is performed and the specified data operations are executed. However, applying the EFSM in passive testing requires the enumeration of data values which is a huge, time and space consuming activity.

In order to easily understand the contributions of this paper, we provide a short outline of our approach. In this paper, the passive testing of a property on a real execution trace integrates two important techniques: symbolic execution of an Input–Output Symbolic Transition Systems and a parametric trace slicing approach. *Input–Output Symbolic Transition Systems* (IOSTS) are commonly used for formally modelling communicating systems interacting with their environment. In IOSTS, the parameters and variable values are represented by symbolic values (called fresh variables) instead of concrete ones. Enumeration of data values is therefore not required. This allows to reduce the huge amount of data values commonly applied in many passive testing approaches. In [10] we proposed this approach to monitor the conformance property alone and then as an improvement in [11] we monitored a property and an attack scenario. A more extended version of our approach is provided in this paper, to specify the protocol properties as well as several kinds of attack patterns. This helps to detect conformance as well as security anomalies.

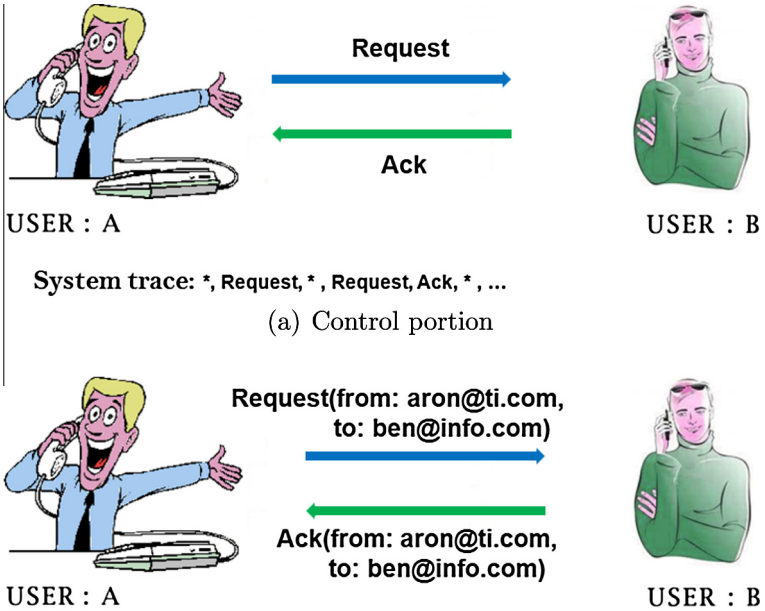

Fig. 1. Property – request followed by acknowledgement response.