Testing Security Policies for Web Applications

Wissam Mallouli, Gerardo Morales and Ana Cavalli
GET/INT, 9 rue Charles Fourier, 91011 Evry Cedex, France
{wissam.mallouli,gerardo.morales,ana.cavalli}@int-edu.eu

***Abstract*—Due to the increasing complexity of Web systems, security testing is becoming a critical activity to guarantee the respect of such systems to their security requirements. To challenge this issue, we rely in this paper on model based active testing. We first specify the Web system behavior using IF formalism. Second, we integrate security rules -modeled in Nomad language- within this IF model using specific algorithms. Then, we perform automatic test generation using a dedicated tool, called HJ2If, developed in our laboratory. Finally, we briefly present a Travel agency system as an ongoing case study to demonstrate the reliability of our framework.**

***Index Terms*—Web Applications, IF Model, Security Policy Specification, Nomad Logic, Test Generation.**

I. Introduction

It is more and more difficult to ensure the respect of Web applications to their security requirements. This difficulty is due to the complexity level of such systems, their variety, and their increasing distribution as well as the high degree of the reliability required for their global functioning. To guarantee such a respect, we need to generate exhaustive test suites including all possible scenarios. To reach this test completeness, we rely in this paper on model based methods.

We first describe the behavioral aspect of the considered Web application under test using IF language [4]. This language is well-adapted to formally describe Web systems features such as hyperlinks, sending and receiving data and client-server communications etc. Second, we specify the system security requirements based on Nomad [6] a security model with 'Non Atomic Actions and Deadlines' that allows to specify without any ambiguity security rules (such as permissions, prohibitions and obligations) in specific contexts with temporal constraints.

The contributions of this ongoing work are (1) the specification of Web application features using IF language, (2) the proposition of a methodology to integrate the security rules within the functional IF model to obtain a secure specification that takes into account the security requirements, (3) and the automatic generation of test cases targeting security constraints. (4) These test cases will be performed in an automatic manner on a real Web system (a travel agency) to check whether its behavior respects its security requirements.

This paper is organized as follows : section II presents the relevant aspects of IF language that permit to formally specify Web systems in a simple way. Section III exposes the methodology to integrate security rules described in Nomad language within an existing IF specification. In section IV, we briefly present our testing methodology approach. Section V illustrates an ongoing case study: a travel agency system. Finally, section VI presents the conclusion and introduces the future work.

II. Modeling Web Systems Using IF Language

Modeling Web applications allows software engineers to specify, understand and maintain their complex architecture in an optimized manner. To perform this formal specification, we rely in our approach on IF formalism [4] which is usually used to model functional behavior of communicating systems such as network protocols and services.

A. Background on IF

The Intermediate Format (IF) language can be considered as a common representation model for other existing languages. It was originally developed to sit between languages as SDL, Promela or Lotos [3] but it has been extended to tackle other notations, as UML [5]. IF is based on communication timed automata and it is used to describe and also validate asynchronous systems.

B. IF Model

IF is a well adapted language to specify Web systems. In particular, based on this language, we can define a set of communicating *processes* to describe different Web application entities and functionalities. The exchanged messages between processes are defined through a set of *signals* with or without parameters depending on different cases. These signals are dispatched using different channels called *signalroutes*. Each process can be instantiated and has his own *identifier* (Pid) and is defined as an Extended Finite State Machine (*EFSM*).

In these EFSM, each *transition* has an enabling condition (*predicate*) and an *action*. Five categories of actions are considered : *assignments*, signal *inputs* and *outputs*, process *creation* and *destruction*. The *urgency* attributes (eager, delayable, lazy) denote transitions priority with respect to the progress of time.

Time constraints are very important in Web systems. The semantic of time in IF language is similar to the one of timed automata. That is : (i) a timed behavior of a system can be controlled through clocks or timers. (ii) The time progress in some state before selecting and executing some transitions. (iii) Transitions take zero time to be executed (instantaneous transitions). All theses properties make it easier to specify Web systems based on IF language.

III. Security Integration Methodology

In this paper, we propose an approach that allow formally testing security rules. This approach manipulates three different inputs: (i) a functional specification of the Web application based on IF language, (ii) a specification of its security policy

(based on the nomad model) and (iii) finally an implementation of the Web system. We want to generate a new specification of the Web system that takes into account the security policy, and then to generate test suite to check whether the implementation of the system conforms to the secure functional specification.

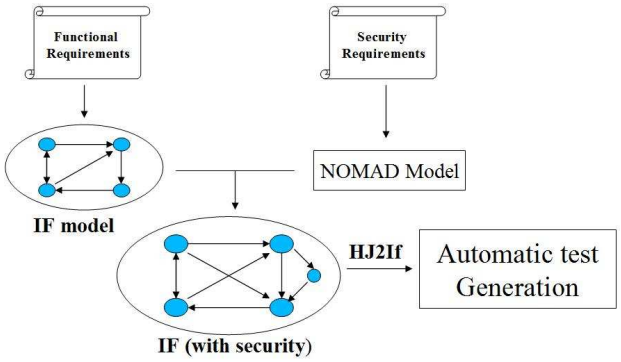

Figure 1. Security Integration into an IF Functional Model

To reach this aim, we have to define algorithms to automatically integrate the security policy rules into the initial specification using a dedicated methodology inspired from [7] where the integration of security rules specified in Or-bac model [2] within an EFSM based specification is presented. The integration process is twofold. At first, the algorithm seeks for the rules to be applied on each transition of the IF specification. Then, it modifies each transition by adding some states, transitions and timers or by updating transition predicates. At the end of the process, this integration will generate a new specification that takes into account the security requirements.

IV. Test Generation

A. Fixing the test objectives

The automatic test generation only target security issues and as a result, it is less time consuming. In fact, if we assume that the unmodified implementation Imp_{In} has been tested and is equivalent to the unmodified specification $Spec_{In}$ and that only transitions that correspond to modified transitions of $Spec_{In}$ are modified in Imp_{In}, we are not obliged to check and test all the scenarios. It is enough to test only the modified transitions which correspond to the target of the designed test objectives.

B. Generation with HJ2If

HJ2If [8] is based on Hit-or-Jump algorithm that is especially used for components testing to perform test sequences generation from the Web system specification. This research is guided by objectives which are illustrated by predicates on transitions (namely stop conditions written in IF language). Research in the partial reachability graphs is performed in depth, width or both at the same time, and is restricted by a limited depth. In order to initialize the generation of test sequences, several parameters are necessary. Four main files must be developed. The first is the Web application IF specification, the second allows the initialization of some variables if necessary, the third one mentions the stop conditions and finally the last one allows the expert to guide the system to a stable state of the system, this file is called 'preamble'. This last one is very important; it allows reducing in a consequent way the length of the test sequence and the duration of its generation. The generated test sequences are usable; they can be produced in TTCN or Message Sequence Chart (MSC) standard notations facilitating their portability. TTCN is a standard language in test domain, while MSC helps to graphically see generated scenarios.

V. Case-study: A Travel Agency

To prove the effectiveness of our framework, we want to carry out a case-study on a travel agency Web application. In our case study, we only consider at first a simple travel agency web application that sells travel tickets and/or bookings in hotels. A payment module is also taken into account in this first step. Thus, a potential traveler can connect to the Web application using a dedicated URL. Then, he/she can request for a ticket to travel to a specific destination or request for a hotel reservation during a specific period. The specification of this travel agency Web application is performed using IF formalism.

Further, we defined some specific security rules to boost the system security. These security rules are inspired from France Telecom security test campaign in the context of POLITESS project [1]. Then, the security rules are formally specified using Nomad model. Their integration will be performed using our algorithms to lead to a secure specification. The generation (using HJ2If) and the execution (using tclwebtest tool) of test cases will be performed in an automatic manner and aim to prove the security respect of our Web travel agency application.

VI. Conclusion and Future Work

In this paper, we presented a framework for active testing of security policies specified in Nomad for Web applications. Our approach consists in automating the integration of security rules within an IF model well adapted to describe Web applications. Afterward, we presented a scheme to derive test cases using HJ2If tool in order to test the conformance of an implementation with respect to its security policy. The Travel agency case study is an ongoing work that will allow demonstrating the reliability of our method.

References

[1] http://www.rnrt-politess.info/.
[2] A. Abou El Kalam, R. E. Baida, P. Balbiani, S. Benferhat, F. Cuppens, Y. Deswarte, A. Miège, C. Saurel, and G. Trouessin. Organization Based Access Control. In *Policy'03*, June 2003.
[3] M. Bozga. Symbolic verification for communication protocols. In *PhD Thesis*, VERIMAG/IMAG, 1999.
[4] M. Bozga, S. Graf, L. Mounier, and I. Ober. IF validation environment tutorial. In *SPIN*, pages 306–307, 2004.
[5] P. Chevalley and P. Thévenod-Fosse. Automated generation of statistical test cases from uml state diagrams. In *COMPSAC*, pages 205–214, 2001.
[6] F. Cuppens, N. Cuppens-Boulahia, and T. Sans. Nomad: A security model with non atomic actions and deadlines. In *CSFW*, pages 186–196, 2005.
[7] W. Mallouli, J.-M. Orset, A. Cavalli, N. Cuppens, and F. Cuppens. A formal approach for testing security rules. In *SACMAT*, Nice, France, 2007.
[8] E. R. Vieira and A. Cavalli. Toward test suite automatic generation with delayable transitions and timing-fault detection. In *RTCSA*, 2007.