AVISPA

www.avispa-project.org

IST-2001-39252

Automated Validation of Internet Security Protocols and Applications

Deliverable 4.6: AVISPA Tool v.3

Abstract

We describe version 3 of the AVISPA Tool for security protocol analysis, focussing in particular on the modifications and improvements, with respect to version 2 of the tool.

Deliverable details

Deliverable version: *v1.0*
Date of delivery: *30.06.2005*
Classification: *public*

Person-months required: *2*
Due on: *30.06.2005*
Total pages: *7*

Project details

Start date: *January 1st, 2003*
Duration: *30 months*
Project Coordinator: *Alessandro Armando*
Partners: *Università di Genova, INRIA Lorraine, ETH Zürich, Siemens AG*

Information Society Technologies

Project funded by the European Community under the *Information Society Technologies* Programme (1998-2002)

Contents

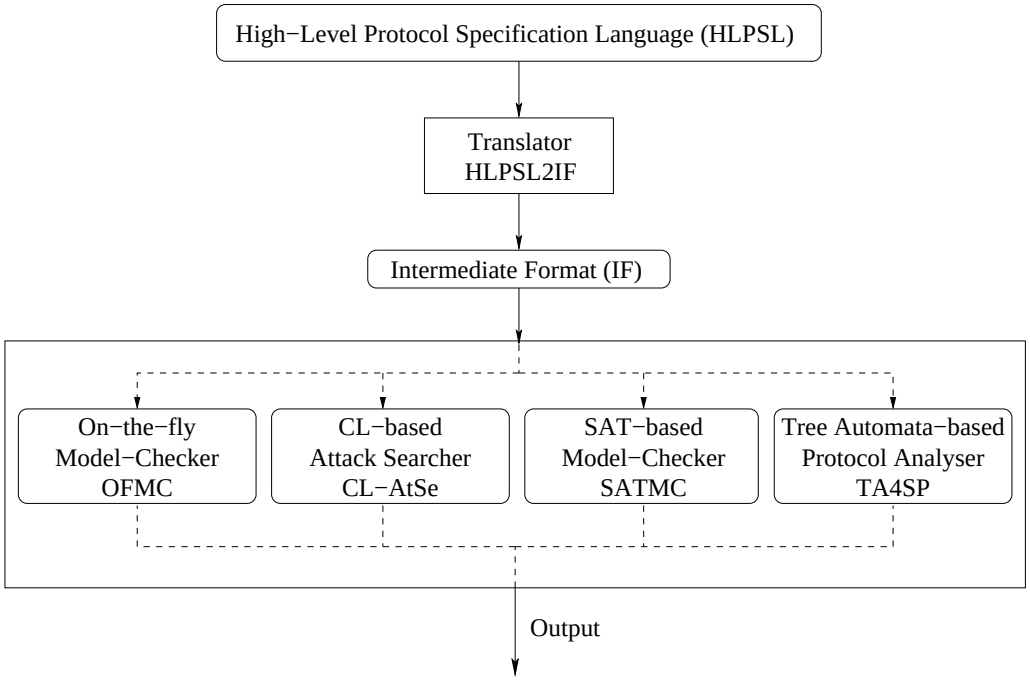

Figure 1: Architecture of the AVISPA Tool v.3

1 Introduction

The architecture of version 3 of the AVISPA Tool for security protocol analysis is depicted in Figure 1. Specifications of security protocols and properties written in the High-Level Protocol Specification Language (HLPSL [1]) are automatically translated (by the translator HLPSL2IF) into IF [2] specifications, which are then given as input to the different back-ends of the AVISPA Tool: OFMC, CL-AtSe, SATMC, and TA4SP. Whenever it terminates, each back-end of the AVISPA Tool outputs the result of its analysis using a common and precisely defined format stating whether the input problem was solved (positively or negatively), some of the system resources were exhausted, or the problem was not tackled by the required back-end for some reason.

In mid-June 2005, we have officially released the AVISPA Tool, which can now be employed by external users thanks to the web-interface accessible from the project web-site, and which is also downloadable as a single "package" to be installed on the users' local machines. Further information is available on the project web-site (`http://www.avispa-project.org`).

This report is organised in the following way. In Section 2 we consider the HLPSL2IF translator. Section 3 is devoted to the description of the main

modifications and extensions that we have carried out on the HLPSL2IF translator and the back-ends with respect to version 2 of the AVISPA Tool It is worth pointing out that the AVISPA Tool v.3 will be thoroughly assessed in a subsequent deliverable [6] by testing the tool against the AVISPA library of Deliverable 6.1 [4], a library of security problems drawn from the protocols developed by the IETF.

2 The HLPSL2IF Translator

The rôle of the HPLSL2IF translator is to automatically translate high-level protocol specifications (written in HLPSL) to low-level specifications (in the Intermediate Format, IF). The HLPSL is a role-based language, designed to be easily readable and writable, that permits one to describe protocols in a precise and formal way; the semantics of this language is based on a fragment of Lamport's Temporal Logic of Actions, TLA [7].

HLPSL supports security protocol specification at a high level of abstraction and has many features that are common to most protocol specifications built-in, such as intruder models and encryption primitives. In contrast, the IF is a lower-level language at an accordingly lower abstraction level; that is, the IF is a tool-independent protocol specification language suitable for automated deduction (so that specifications in the IF can be given as input to the analysis back-ends of the AVISPA tool).

Further information on HLPSL and IF, including their syntax and semantics, can be found in the user manual and in the tutorial included in the AVISPA Tool distribution (`http://www.avispa-project.org`).

We have upgraded our translator from HLPSL to IF according to the recent language extensions. The major improvements concern:

Syntax In HLPSL, we have improved the syntax in order to support a more intuitive representation of assignment and freshness.

Sets The handling of sets has been improved; deletion of an element is now possible.

Goals The HLPSL *goal section* has been simplified: macro goals refer to labels; LTL goal formulae can now be specified; and each *secret* goal refers to a constant set of agents. In IF, two goal sections are generated: one representing LTL properties to be checked; one representing attack states to be detected.

In addition to those improvements, the type *list* has been removed (rarely used and not completely handled).

The translator is invoked by typing:

```
hlpsl2if [options] [inputFile.hlpsl]
```

where `options` is a combination of:

`--types`	for printing identifiers and their types,
`--init`	for printing the initial state,
`--rules`	for printing the protocol rules,
`--goals`	for printing the goals,
`--all`	for printing everything (default),
`--split`	for generating one IF file per goal,
`--stdout`	for printing on the standard output,
`--output dir`	for setting the output directory (default: same as input),
`--nowarnings`	for not displaying warnings,
`--version`	for printing the version number,
`-help`	for printing this list of options, and
`--help`	for printing this list of options.

3 The Back-Ends of the AVISPA Tool v.3

In this section, we briefly describe the main modifications and extensions that we have carried out on the back-ends of the tool during the final 6 months of the project, in order to further optimise and consolidate them.

3.1 OFMC

During the last 6 project months, we have improved the performance of OFMC by removing redundancies in the search procedure. We have also been able to identity and correct a number of minor bugs, thanks to the extensive experimentation with the large number of protocols that the AVISPA library now provides.

3.2 CL-AtSe

During the last 6 project months, we have worked intensively to integrate new algebraic properties in CL-AtSe. In the first place, a complete unification algorithm modulo XOR has been integrated (this algorithm is a specially adapted version of the generic Baader/Schultz unification algorithm). Moreover, this algorithm was implemented in a modular way to make future integration of other theories easier. Following the same direction, a second

unification algorithm has been implemented, this time dedicated to the algebraic properties of exponentiation. While this second algorithm is not fully complete yet, it is complete enough to find interesting attacks on our test-suite. Along with these algorithms, we integrated new intruder deduction to allow the intruder to benefit from the properties of these new algebraic operators.

Separately from this work, we also carried out a considerable amount of work on the tool kernel to improve the existing algorithms and implement all the changes decided globally by the AVISPA consortium. In particular, the attack detection methods were modified to correspond precisely to the IF semantics.

Finally, some efforts were also spent on CL-AtSe's verbose output to improve its readability, thereby giving the end-user more information both on the protocol specification, once translated and unrolled, and on the attack found (if any).

3.3 SATMC

During the last 6 project months, we have improved the interface and the underlying techniques of SATMC. Moreover, thanks to the intensive testing phase performed by us and by external users, we were able to identify some bugs, which we have fixed. This has lead to a new version of SATMC that is simpler to be invoked and more stable than the previous one.

3.4 TA4SP

During the last 6 project months, we have updated the TA4SP back-end according to the HLPSL changes about secrecy declarations. Before, a secret data X shared between two agents A and B was denoted by the conjunction of the two predicates $secret(A, X)$ and $secret(B, X)$. The new secrecy declaration needs sets of agents in order to express the idea that a data is shared a set of agents. Thus, $secret(A, X) \bigwedge secret(B, X)$ becomes $secret(X, secrecy\_identifier, A, B)$. Even if TA4SP does not support sets in general (due to insertion or deletion of an element and tests done on sets), we have carried out an extension to support only the sets used by secret facts (this extension is possible since these sets are considered as constants, i.e. without alterations).

4 The Output Format

In order to facilitate the automatic parsing and gathering of the results yielded by each back-end, the output language has undergone a joint standardisation effort carried out by all project partners. The result of this process is the AVISPA Output Format (OF), intended to provide a clean and human-readable, yet uniform and well-defined, representation of the output of each back-end.

Currently, the back-ends of the AVISPA Tool output the following information:

- the result obtained by the back-end — that is, whether the input problem was solved (positively or negatively), or the problem was not tackled for some other reason;
- optionally, some statistics about the required resources (e.g., depth of the search, internal timings etc.);
- a description of the considered goal and, in case it was violated, the related attack trace.

The OF has not undergone any major changes: it allows for different visualisations of the attack traces as message-sequence charts in the graphical user interface of the AVISPA Tool, where the users will also be able to specify/edit their own protocols in the HLPSL input language, and it also allows for the generation of corresponding postscript files of the attacks.

References

[1] AVISPA. Deliverable 2.1: The High-Level Protocol Specification Language. Available at `http://www.avispa-project.org/publications.html`, 2003.

[2] AVISPA. Deliverable 2.3: The Intermediate Format. Available at `http://www.avispa-project.org/publications.html`, 2003.

[3] AVISPA. Deliverable 4.4: AVISPA tool v.1. Available at `http://www.avispa-project.org`, 2003.

[4] AVISPA. Deliverable 6.1: List of selected problems. Available at `http://www.avispa-project.org`, 2003.

[5] AVISPA. Deliverable 7.3: Assessment of the AVISPA tool v.2. Available at `http://www.avispa-project.org`, 2004.

[6] AVISPA. Deliverable 7.4: Assessment of the AVISPA tool v.3. Available at `http://www.avispa-project.org`, 2005.

[7] L. Lamport. The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, 16(3):872–923, May 1994.