UML-Based Modeling and Formal Verification of Security Protocols

S. Mota**, B. Fontan*,**

*ENSICA, 1 place Emile Blouin, 31056 Toulouse Cedex 05, France
**LAAS-CNRS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 04, France
smota@laas.fr; bfontan@ensica.fr

Introduction

Security protocols are typical examples of algorithms which deserve the use of formal methods for their analysis against intruder attacks.
Among security protocols, key management protocols for group communication are very complex for formal verification because of their specific constraints (dynamicity, time, reliability and number of users). The difficulty to guarantee a correct behaviour of the protocol is increased proportionally with the strength of these constraints.
This paper presents an ongoing work on modeling and formal verification of a secure group communication protocol using TURTLE [APR04], a UML profile based on the formal language RT-LOTOS. The protocol [HAS 05] under verification is being developed in SAFECAST [SFC], a project funded by the French RNRT research network.

I TURTLE

TURTLE (Timed UML and RT-LOTOS Environment) [APR04] is a UML profile supported by the TTool toolkit that includes a chain of three tools: a RT-LOTOS code generator, a RTL (RT-LOTOS laboratory) [RTL], and an interface to CADP-Aldébaran. TTool also enables the edition of TURTLE diagrams.
The TURTLE profile has been defined for the specification, design and verification of real-time distributed systems.
RTL is used to validate formal RT-LOTOS specifications. These specifications are automatically generated from TURTLE models by a code generator. CADP-Aldebaran minimizes the reachability graph output by RTL. TTool invokes RTL and CADP-Aldebaran via a user-friendly interface, and makes the use of the formal language RT-LOTOS transparent to the end user.

II Verification methodology of security protocols

Verification by abstraction is the technique used to study the modeled systems in TURTLE. In the verification by abstraction a Reachability Graph is obtained to observe all possible executions of one system. The most important paths (frequently the paths in which errors could occur) have to be manually checked to validate the system.
This path observation is friendly since it permits a graphical presentation of each result, but incomplete since there is no guide to lead the observation. The user does not have the capability of using constraint solver tools. Then the search of inconsistencies is not rigorously made.
A reduced reachability graph called minimized graph or quotient automaton can be obtained by minimization by abstraction. In this technique the user establishes observing points and obtains the equivalent minimized graph in order to facilitate the analysis of the behaviour of the system. Nevertheless the search of errors is still informal.
Then a methodology to obtain a “good” quotient automaton is explained in this publication. The generation of this quotient automaton is guided by the establishment and satisfaction of properties.

Proposing and using a verification methodology based on observers is based on a three steeps UML_modeling process. The methodology is shown in fig. 1.
The first step in the modeling process is the protocol analysis; the system to verify is described by UML diagrams. Use case diagrams and sequence diagrams are available in TURTLE to support in the analysis phase.
The second step consists in the description of the architecture using class diagrams, and in the specification of the behavior with comportment diagrams.
Finally, in step 3 the methodology for the verification of the security protocols is developed. The goal is to automatically guide the observation of the system behavior with respect to the properties to be checked and the finding of paths in the reachability graph.
To do this, the definition of adequate observers is proposed.
Even if TURTLE produces the reachability graph of the protocol with the laboratory RTL and if a minimization of this graph can be made with a friendly interface of the CADP (Aldebaran) tool, the introduction of observers where the properties to satisfy are modeled and introduced is necessary. The creation of these observers is conducted by an external user.

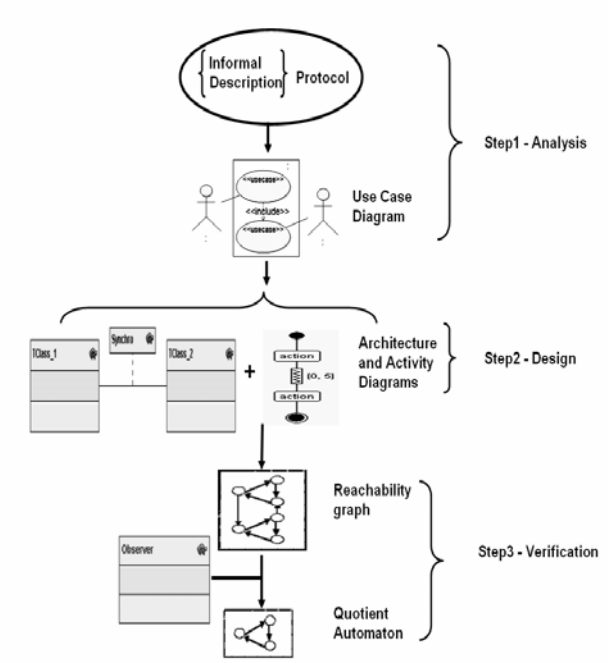

Fig 1. Methodology of verification of security protocols

Finally the formal verification is accomplished with the results observed in the quotient automaton that results from these observers. The quotient automaton shows whether the properties are fulfilled or gives the inconsistence in the system for these properties.
This type of observers avoids a quite long manual analysis

The first case study of the proposed methodology has been the well-known Needham-Schroeder Public-Key-Protocol (NSPK) [LOW 96]. The classical Man_in_the_middle attack found by Gavin Lowe has been easily identified and isolated by the observer of the verification methodology.

III Ongoing work: SAFECAST, a secure group Communication

Specification, modeling and verification of secure group communication protocols are the next step in our work. This joint research activity between LAAS, LORIA, EADS, ENST and UTC is supported by the French RNRT project: SAFECAST [SFC].
The SAFECAST project aims at designing secure group communication protocols for Private Mobile Radio applications (PMR) [HAS 05]. For this purpose, it aims at validating the group communication protocols to verify that they enforce the defined security properties and requested performances.
This project designing security protocols will be the support of theoretical studies for security requirement in group communication applications. These results will be the first steps towards the standardisation of this type of applications.
The complexity of security in group communications comes from the inherent factors attached to security requirements and also from the group characteristics: number of users, dynamicity and structure changes. In this project, the possibilities of radio and ad-hoc networks are considered, including reliability and security properties. High dynamicity and hierarchical organisation will define the group administration strategy that will be related to time constraints [HAS 05].

Groups of hundreds of users are assumed to stay in communication during intervals of many hours.
Each group will have a hierarchy assigned by a role called class, with one leader in each group.
Three types of group functions are defined:

- Intra group: Activities inside the group like join, leave, exclude, go to upper class and go to lower class.
- Group Management: Activities dedicated to keys management. Keys will be used to guarantee security services during group communication.
- Inter group: Activities between groups as merging and splitting groups.

A 3-layers architecture is under development:

- Broadcast PMR communication: It is the distribution physical layer available in PMR networks (Communication point to point is also considered).
- Group Management layer, which includes the designed protocol.
- Users with roles: this is the application layer.

The correct behavior of the system will be checked, because services as member authentication, source authentication, information confidentiality, integrity and reliability, have to be guaranted.
Finally, time restrictions during group activities are also considered.

IV References

[APR04] L. Apvrille, J.-P. Courtiat, C. Lohr P. de Saqui-Sannes, “TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit”, IEEE Transactions on Software Engineering, Vol. 30, No. 7, July 2004, pp.473-487.

[SFC] http://safecast.loria.fr/

[HAS 05] H.R. Hassan, A. Bouabdallah, H. Bettahar, Y. Challal; “Hi-KD: An Efficient Key Management Algorithm for Hierarchical Group Communication”, IEEE-SecureCom’05 (Greece).