On-the-fly Model Checking of Security Protocols

Guoqiang Li
Japan Advanced Institute of Science and Technology

1 Aims

Due to extreme importance of security protocols in the modern world, to ensure security protocols to be more reliable, and to speed up the development of new protocols, it is important to have tools that support the activity of finding flaws in protocols, or guaranteeing correctness of protocols. Various formalisms for the specification of security protocols have been proposed based on, e.g. process calculi, belief logic, higher-order logic, first-order logic, linear logic, equational logic, hidden algebra, and tree automata. They adopted different full-automatic or semi-automatic techniques, such as model checking based on finite models, resolution, and higher-order theorem proving. However, security protocol analysis proves to be a rather difficult task. Due to the complication of the network, models for security protocol analysis should be carefully designed to depict each infinity factor one assumed.

A dilemma has occurred when one tries to propose a methodology for security protocol analysis, dealing with the infinity factors above. On the one hand, a model for security protocols should be strong enough in expressiveness to describe any possible situation that a running protocol may reach, or it may fail in detecting subtle attacks of a security protocol. On the other hand, analyzing a property on a model strong in expressiveness may be undecidable. Thus automatic techniques may not terminate when detecting flaws.

Our aim is to propose a flexible and expressive model to represent behaviors of security protocols. Deductive systems can be inserted freely to integrate the model, to represent infinitely many messages intruders or dishonest principals generate, due to different security assumptions. Driven by different security requirements, various security properties can be analyzed in the model. The analysis method is sound and complete with respect to the analyzed property, under the assumption provided by the model. The objectives are listed as follows,

- to design a flexible expressive framework suited for analyzing various security properties under reasonable assumptions;
- to develop a fully automatic tool for security protocols analysis, sound and complete under certain assumptions. That is, when flaws are not detected, it guarantees that a protocol is secure under these assumptions;
- to find out a method to generate a specification from a protocol description automatically, making the tool easily usable.

2 Approaches

To describe behaviors of each principal in a security protocol, we choose a process calculus based on a variant of Spi calculus, in which new syntax, the *binder*, and a new primitive that binds variables, *new*, are introduced. A variable bound by the new primitive will range over a given name set. In the process calculus, we use *identifiers*, instead of *replications*, to describe infinite operation in a process. We choose to use identifiers, since with a *sequential* restriction (inspired by CCS) upon a process with recursive processes, this process can be naturally described and analyzed by the *pushdown system*.

To model a hostile network, the calculus uses environment-based communication, instead of the standard channel-based communication. Principals exchange the messages with the environment. Following the Dolev-Yao model, the environment can store, duplicate, hide or replace messages that travel on the network. It can also operate according to the rules followed by honest principals and synthesize new messages by pairing, decryption, encryption and creation of fresh nonces and keys, or by arbitrary combinations of these operations. Thus a principal waiting for an input at a given moment may expect any of the infinitely many messages the environment can produce and send in the network.

3 Achievements

We adopt model checking as the execution engine, and introduces a process calculus based on a variant of Spi calculus to describe behaviors of a security protocol. An environmental deductive system are introduced to generate messages that intruders can send. The main achievements in this year are as follows.

Model checking of Recursive Protocols. A sound and complete model checking is proposed to analyze *recursive protocols*. It makes the first step towards model checking of security protocols in an unbounded number of sessions. To the best of my knowledge, this is the first model checking method applied to recursive protocols. An attack for the protocol is first detected automatically in our framework. It is also the first infinite model checking method applied to security protocol analysis.

Model Checking Non-repudiation and Fairness in Bounded Sessions. By the introduction of another deductive system to describe the capability of dishonest principals in the model, security properties, such as non-repudiation and fairness, of *fair non-repudiation protocols* in bounded sessions can be analyzed by a similar parametric approach. To the best of my knowledge, this is also the first model checking method applied to the non-repudiation property.

Protocol-Independent Security Specifications. Protocol-independent specifications for secrecy and authentication properties are proposed. In this approach, the specifications for secrecy and authentication properties can be generated automatically from a protocol description. In comparison, other formal methods for security protocols, especially process calculi based approaches, manually generate a security specification dependent on a given protocol. Our choice is to scan a formal protocol description statically by a syntax scanner, then the process for a security specification, named a *probing process*, is generated automatically from a formal protocol description.

Implementation by Maude Maude, a language and system supporting both equational and rewriting logic computation for a wide range of applications, is chosen to testify our methodology. By the facility of the reachability analysis in Maude (implemented as `search`), each property can be checked at the same time while a model is being generated. Hence it is named *on-the-fly model checking* methodology.

4 Future work

The first future work will adopt the method for other properties. For instance, anonymity, and other authentication variations can be analyzed within the framework. Another choice is to analyze existing security properties for other kinds of security protocols, such as, fairness for electronic commerce protocols, or certified e-mail protocols, and so on. The fairness for these protocols have slight differences from what we have defined for fair non-repudiation protocols. In addition, security properties for security protocols with multiply parties, etc.

A translator that translates a formal protocol description to a Maude source file is designable. It will also contain the automatic transformation approach to generate a security specification for a given security property.

Refinements should be applied to the current tool, in order to analyze more complex, practical protocols. For instance, to implement a more efficient unification algorithm (The analysis relies heavily on unification, while the current algorithm is quite time-consuming), and to refactor current analysis steps.

5 Publication

- Guoqiang Li, Mizuhito Ogawa. On-the-fly model checking of security protocols and its implementation by Maude. IPSJ Transactions on Programming, Vol.48, No. SIG 10(PRO 33), 50-75, 2007
- Guoqiang Li, Mizuhito Ogawa. On-the-fly Model Checking of Fair Non-repudiation Protocols. In Proceedings of the 5th International Symposium on Automated Technology for Verification and Analysis (ATVA'07), LNCS 4762, 511-522, 2007