2010 International Conference on Economics, Business and Management
IPEDR vol.2 (2011) © (2011) IAC S IT Press, Manila, Philippines

On the Design of Communication in Business Information Systems

Pavel Ocenasek
Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic
e-mail: ocenaspa@fit.vutbr.cz

***Abstract*— This paper presents a novel approach that has been used for designing secure communication in the business information systems. The approach is based on the evolutionary approach and uses principles of genetic algorithm to "evolve" the communication protocols. The protocols can be than used for establishing secure communication as well as for client/server authentication and/or key distribution in information systems.**

Keywords- Business information system, communication, design, security protocol, evolution

I. INTRODUCTION

The increasing popularity of distributed computing and applications like internet banking and electronic commerce has created both tremendous risks and opportunities. Many of risks stem from security breaches, which can be ruinously expensive. One of the cornerstones of security is the use of security (cryptographic) protocols in which information is exchanged in a way intended to provide security guarantees.

Security protocols are becoming widely used and many new protocols are being proposed. Since security protocols are notoriously difficult to design, computer assistance in the design process is desirable.

In this paper we describe the goals of security protocols – user authentication and/or key distribution and we propose an evolutionary technique that may be useful in designing new security protocols.

II. SECURITY PROTOCOL

A security protocol is a recipe that describes the operations in which the subjects should achieve some security goals. Protocols are often described using informal notation, for example as a sequence of instructions explaining the actions taken by the subjects.

Each step describes an event $A \rightarrow B: X$, which states that A exchanges the message X with B. Messages consist of atoms, like subject names and nonces (randomly generated strings), and can be composed. Moreover, messages may be encrypted using keys of subjects. Because security protocols may contain certain flaws, finding such attacks is the purpose of formal validation using various approaches [1] [2] [5].

A. Introduction

A protocol is a recipe that describes how subjects should act to achieve some goal. Protocols are often described using informal notation, for example as a sequence of instructions explaining the actions taken by the subjects.

Each step describes an event $A \rightarrow B: X$, which states that A exchanges the message X with B. Messages consists of atoms, like subject names and nonces (randomly generated strings), and are composed by tupling. Moreover, messages may be encrypted using keys of subjects.

However, describing the protocol components, cryptographic properties and requirements in details is beyond the scope of this abstract. For more information please refer e.g. to [6].

B. Needham-Schroeder Protocol

Probably the best known security protocol used for authentication is the Needham-Schroeder protocol (in this example we use the public-key version of this protocol):

1. $A \rightarrow S$: A, B
2. $S \rightarrow A$: $\{ K_B, B \}_{K_S^{-1}}$
3. $A \rightarrow B$: $\{ N_A, A \}_{K_B}$
4. $B \rightarrow S$: B, A
5. $S \rightarrow B$: $\{ K_A, A \}_{K_S^{-1}}$
6. $B \rightarrow A$: $\{ N_A, N_B \}_{K_A}$
7. $A \rightarrow B$: $\{ N_B \}_{K_B}$

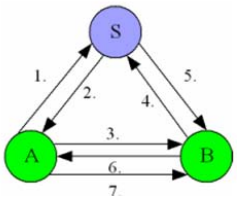

Figure 1. Needham-Schroeder Protocol.

This protocol can be considered as the interleaving of two logically disjoint protocols: messages 1, 2, 4 and 5 are concerned with obtaining public keys, whereas messages 3, 6 and 7 are concerned with the authentication of A and B.

The protocol was found unsecured by Lowe and new fixed version was published. The proposed attack allows an intruder to impersonate another agent.

III. MOTIVATION

There are five primary goals of the research:

1) Modification of standard evolutionary techniques for the use in the field of security protocol design. Designing new protocols is a completely different problem, other than those previously solved by these techniques. Protocols are programs with instructions that are evolved and verified for security.

2) Optimization of fitness function calculation that is used to evaluate the protocol. Calculating the fitness function is a crucial point in the evolutionary process. The value will be computed from the results produced by external security verification tools. In principle, the fitness

corresponds to the percentage of met security presumptions, e.g. the protocol length, satisfactions of initial requirements (sets of knowledge and belief) and security properties.

3) Optimization of the generation of elementary instructions and message components used for representing security protocols. The number of elementary instructions and components is another crucial point. High number of invalid instructions increases the size of search space. Lower number of instructions and components increases the design performance.

4) Analysis of the heuristic strategies that can be used in the design process. The evolution process itself is not powerful enough to efficiently explore the space of possible protocols. Additional heuristics might be needed to accelerate the evolution process, e.g. the heuristic crossover operator, heuristics for fitness computation, etc.

5) Design and implementation of an application, evaluation of the performance of proposed approach and the influence of proposed heuristic techniques.

This research aimed to address the shortcomings of existing approaches, namely to minimize the requirement of human-intervention. So the general goal is to make the protocol design process highly automated.

IV. PROTOCOL DESIGN

As mentioned above, the protocol is a set of rules and conventions that define the communication framework between two or more subjects. These rules may be elementary instructions that consist of operations such as sending a message, encrypting/decrypting a message with a secret key. This chapter proposes a novel algorithm for automated design of communication and security protocols.

A. Protocol Enconding

The very important task was to choose the right and efficient form for protocol representation. Security protocols are encoded into chromosomes as strings of elementary instructions. Each chromosome (individual) in population represents one random protocol.

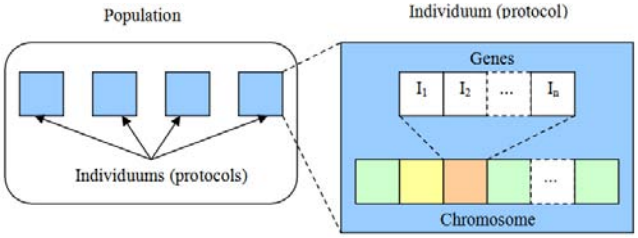

Figure 2. Structure of Population. Protocol (individuum) is encoded as a sequence of instructions.

The the chromosomes, unlike in standard genetic algorithms, may be of variable length. This is due to the optimization of protocol instructions.

B. Algorithm

While the protocol runs, each instruction affects the corresponding sets of knowledge and belief, described using BAN logic.

From the evolutionary-optimization point of view, a chromosome of variable length represents the sequence of protocol instructions. There could be generated as many random chromosomes as required for initial population. Each chromosome represents a different protocol and its fitness is computed by simulating its run, according to the changed sets of knowledge and belief.

1) Security goals

Before we start the design of new security protocol, we need to state the initial presumptions and security goals to reach. We should specify what should or shouldn't contain sets of knowledge and belief for each involved subject, desired protocol length, forbidden knowledge for each involved subject, etc.

Some of the initial requirements are set in the generator, e.g. which message is secret (implicitly keys are considered as secret) and cannot be sent unencrypted via unsecured channel, etc.

Some of the security goals can be specified within the grammar, e.g. format of protocol instructions, number of recursive encryptions and some specific security properties.

The main algorithm (see figure below) derives from the genetic algorithm and describes the whole flow we use to design security protocols. The algorithm is described in detail in the following paragraphs.

2) Initial Population

Randomly generated protocols (instruction sequences) are encoded into the chromosomes. In this step the fitness of all individuals is calculated.

The intelligent generator creates a random population of protocols using the BNF form (Backus Naur Form) of context free grammar. The generated protocols are valid, so they are executable. The generator considers the security of channels and properly maintains the sets of knowledge for each subject: the subject is able to send only messages stored in his set of knowledge.

In this research we use the presented grammar in our examples for creating common security protocols.

3) Evaluation

In this step, the evaluation of each protocol is performed. The fitness value depends highly on the presumptions and satisfaction in each state of the protocol run. The employment of some additional verification tools for finding certain flaws might be helpful. To calculate the fitness of solutions - generated security protocols - the instruction sequences must be traced to simulate their outcomes. In security protocols, we use various measures to quantify the quality of a generated protocol. The higher is the fitness, the less security flaws exist and the more initial presumptions are satisfied.

The fitness is computed (implicitly) from the following parameters:

- required / received knowledge,
- required / received belief,

- protocol length,
- duplicate instructions,
- forbidden instructions,
- invalid instructions

4) Selection

Like in standard genetic algorithms, the individuals with the best fitness are stochastically chosen to be parents for mating. For the proposed approach we have tested various selection operators for protocols. The best seemed to be the roulette wheel selection.

5) Crossover

The choice of the right crossover operator and locations in chromosomes is very important. It may highly affect the chances for generating a chromosome with better fitness. The basic idea for mating is that two chromosomes may cross at selected states (instructions) if both have corresponding sets of knowledge and belief. This means that we have to prove that after crossing the instruction strings, the rest of protocol makes sense for both individuals.

6) Replacement with offspring

The produced individuals are replaced to the new population and the evolution process starts over again from the step "Choosing parents".

The whole design is finished when some individuals (with best fitness) satisfy the initial presumptions or the maximum number of generation is reached.

7) Finishing the evolution

The whole design is automated and evolves according to the algorithm specified above. The result is the chromosome with the best fitness (that mostly satisfies security requirements), which can be interpreted as a sequence of instructions in the security protocol. The final optimization is performed among the resulting solutions. This means generally that the duplicate instructions (instruction that perform the similar operations) are removed.

Although we may obtain during the evolution solution with the minimum protocol length (as the length parameter is considered in the fitness calculation too), the minimization would be performed during a very long process of evolution and thus we could obtain the protocols not clearly minimized. This is the main reason for performing the final optimization separately.

In practice, it is valuable to consider as the result the top-n protocols (e.g. $n = 10$) and verify them using some external verification tools. This is due to the principles of evolutionary approach, where the solution with the best fitness isn't always the best solution in the real-world conditions. Although the solution may have the best calculated fitness, it might contain several security flaws.

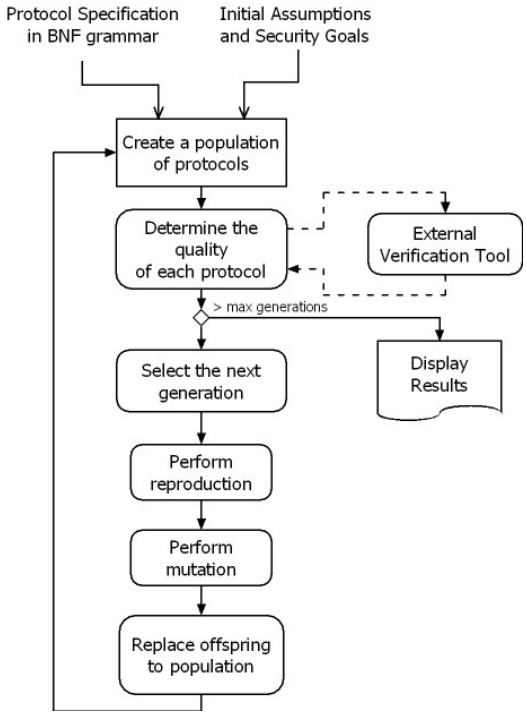

Figure 3. Evolutionary Algorithm for Security Protocol Design.

C. Summary

As we can see, the process of protocol design can be highly automated. The use of this approach is desirable especially in the design of complex protocols where e.g. many subjects are involved.

The crossover and mutation operators have to be modified for generating simple security protocols as well as the process protocols evaluation process. The usability for more complex protocols requires addition of heuristic strategies (e.g. implemented intelligent instruction injection). Although the security requirements are specified at the beginning of the automatic generation, the use of additional verification techniques might be useful to check other security properties and the protocols correctness.

V. PRACTICAL RESULTS

Example: Security Protocol with Authentication (Symmetric Version)

Let us consider subjects A, B and S and we have to generate a symmetric-key security protocol for authentication of subject A to the subject B (with the nonce N_B generated by subject B), using the third trusted party – server S.

A. Intial Assumptions and Goals Specification

The initial sets of knowledge and beliefs for each involved subject should be specified as well as the goals to be reached, communication channels and evolution parameters. The sets of knowledge represent the information each subject knows (learns during the communication). The

sets of belief represent the information each subject assumes that other subjects know (learn during the communication).

B. Evolution of Protocol Population

The design process was performed on 30 different runs. Each run represents an evolution of population (300 protocols) over 200 generations.
Statistical information:

- Number of runs: 30
- Standard deviation of the fitness value for best-candidates from each run: 0,396

C. Resulting Protocol

Optimized version of generated protocol:

1. $B \rightarrow A : N_B$
2. $A \rightarrow B : \{N_B\text{-}1\}K_{AS}$
3. $B \rightarrow S : \{N_B\text{-}1\}K_{AS}$
4. $S \rightarrow B : \{N_B\text{-}1\}K_{BS}$

Final optimization of resulting protocol consists of removing the duplicate instruction (if any) and in removing instruction that do not affect the sets of knowledge and belief of involved subjects.

As we can see, the resulting optimized protocol satisfies the goals for knowledge and belief. The subject *B* generates nonce N_B and sends it to the subject *A*. Subject *A* receives the nonce, computes the equivalent $N_B\text{-}1$ and sends it back to the subject *B*, encrypted with the only key that subject A knows: K_{AS}. Subject *B* receives this message and forwards it to the trusted subject *S*. This subject *S* decrypts the message, encrypts it with the "proper" key K_{BS} and sends it back to the subject *B*. This means that subject *B* receives $N_B\text{-}1$ which was computed only by the subject *A*. In this way the subject *A* proved the knowledge of N_B and the knowledge of K_{AS} respectively.

VI. Conclusions

There are many areas of practical applications of security protocols. These protocols are used in all the communication between subjects, where the authentication process or key establishment is needed. The security protocols are widely used in the computer network communication (e.g. SSL protocol) where they serve for establishing the encrypted communication between server and client (commonly used in the area of electronic banking – e.g. client and bank). The other usability is in the area of embedded systems or electronic commerce.

Acknowledgment

The research has been supported by the Czech Ministry of Education in frame of the Research Intention MSM 0021630528: Security-Oriented Research in Information Technology, MSM 0021630503 MIKROSYN: New Trends in Microelectronic Systems and Nanotechnologies, by the Grant Agency of the Czech Republic through the grant GACR 102/08/1429: Safety and security of networked embedded system applications and by the project FIT-S-10-2: Recognition and presentation of multimedia data.

References

[1] Abadi M., Needham R.: Prudent Engineering Practice for Cryptographic Protocols. In: Proceedings of the 1994 IEEE Symposium on Security and Privacy, IEEE Computer Security Press (1994) p. 122-136

[2] Abadi, M., Tuttle, N.: A Semantic for a Logic of Authentication. In: Proceedings of the ACM Symposium on Principles of Distributed Computing (1991) p. 201-216

[3] Agray, N., van der Hoek, W., de Vink, E.: On BAN Logics for Industrial Security protocols. CCEMAS (2001) p. 8

[4] Burrows M., Abadi M., Needham R.: A Logic of Authentication. ACM Transactions on Computer Systems, 8 (1) (1990) p. 18-36

[5] Gritzalis S.: Security protocols over open networks and distributed systems: Formal methods for their Analysis, Design and Verification. University of Aegean, Greece, Computer Communications, 22 (8) (1999) p. 695-707

[6] Ma, L., Tsai, J.: Formal Verification Techniques for Computer Communication Security Protocols. Handbook of Software Engineering and Knowledge Engineering, World Scientific Publishing Company (2000) p. 23

[7] Očenášek, P., Trchalík, R.: Modal Logics Used for Authentication Protocols Analysis: Survey and Comparison, In: Proceedings of the 7th International Carpathian Control Conference, Ostrava, CZ, VŠB-Technical University of Ostrava (2006) p. 401-404

[8] Očenášek P., Očenášek J.: Designing Secure Communications Using Evolutionary Approach, In: Genetic and Evolutionary Computation Conference GECCO 2006, Seattle, WA, US, ACM (2006)

[9] Očenášek, P.: Towards Selected Problems in the Security Protocol Design and Verification. 1st Doctoral Workshop on Mathematical and Engineering Methods in Computer Science MEMICS, Znojmo, CZ (2005)

[10] Shmatikov, V., Stern, U.: Efficient Finite-State Analysis for Large Security Protocols. In: Proceedings of the 11th IEEE Computer Security Foundation Workshop, IEEE Computer Society Press (1998) p. 10

[11] Alves-Foss, J., Soule, T.: A weakest precondition calculus for analysis of cryptographic protocols, DIMACS Workshop on Design and Formal Verification of Crypto Protocols (1997).