# DESIGN OF CRYPTOGRAPHIC PROTOCOLS BY MEANS OF GENETIC ALGORITHMS TECHNIQUES

Luis Zarza, Josep Pegueroles, Miguel Soriano, Rafael Martínez
*Jordi Girona 1-3 CAMPUS NORD C3 08034 Barcelona Spain*

Keywords: Genetic algorithms, Cryptographic protocols, Automatic protocols design, Protocols evaluation.

Abstract: Genetic algorithms techniques are broadly accepted as an easy way to solve optimization problems. They provide, in a reasonable time, optimal or near-to-the-optimal solutions to problems involving a large amount of variables and entries. In this work we present Genetic Algorithms as a tool aiding the design of security protocols. The design process is divided in the following steps: a population consisting in a set of protocols is established; the population evolves according the benefits criteria programmed in the evolution process. The mapping of valid protocol messages to individuals in a population and the election of proper genetic algorithm evolution mechanisms are presented as key items in the whole process. All proposals in this work have been implemented in a software tool including basic features as cryptographic protocols design using public key and symmetric cryptography. Results achieved with simple examples confirm our expectations and point as future work the development of new versions including advanced features.

## 1 INTRODUCTION

The genetic algorithms, based in Charles Darwin's Evolution Theory, constitute a problem solving technique with great acceptance. However, its application to the design of cryptographic algorithms has not been studied enough. In this paper we introduce some of the fundamental concepts required for its understanding and application, and its utility and power to solve security problems are shown.

Most of the genetic procedures are relatively easy to implement. Generally the most difficult aspect to achieve is the evaluation of each individual. Generally speaking, it's assumed that it is not hard to find a good solution to a problem if there is a procedure to quantify and evaluate different solutions for that problem. In such conditions even the optimal solution can be easily found. On the contrary, when problems to solve refer to systems with large amounts of variables, or worse, when formal mathematical methods are not available for modeling requirements and optimize solutions, the finding of a mechanism to evaluate different proposals or solutions to the problem is very useful. In this sense, we propose the use of Genetic Algorithms to the design of complex cryptographic protocols, including many variables and with non existing formal evaluation method.

The evaluation of security protocols must consider fundamental aspects such as required amount of data for transmission or the prevention of malicious parties accessing it, among others.

## 2 GENETIC ALGORITHMS (GA)

J. H. Holland established the principles of genetic algorithms, inspired by the book "The Genetic Theory of the Natural Selection", from the evolutionary biologist R. A. Fisher. Holland started its work in 1962 presenting it them in the book "Adaptation in Natural and Artificial Systems", published in 1975 (Holland, 1975). Holland proposed to face complex problems acting as natural systems do with the adaptation processes for evolution. GA is a metaheuristic technique, and the basic idea behind its proposal was to design artificial systems that can replicate as natural beings. Each element in a population would represent a possible (not optimal) solution, and the generation of new elements would be done according to the "natural selection" rules in their way to optimality.



Zarza L., Pegueroles J., Soriano M. and Martínez R. (2006).
DESIGN OF CRYPTOGRAPHIC PROTOCOLS BY MEANS OF GENETIC ALGORITHMS TECHNIQUES.
In *Proceedings of the International Conference on Security and Cryptography*, pages 316-319
DOI: 10.5220/0002102003160319

## 2.1 GA and Optimizing Problems

The most common application of genetic algorithms has been the solution to optimization problems, where have been shown to be very efficient. Nevertheless, it's not possible to apply GA techniques to any optimization problem; next the most important requirements a problem must satisfy to be solved with this techniques are: the optimal solution must be among a finite number of possible proposals enumerated, an aptitude function for evaluating proposals must be defined, and the proposals must be able to be codified so that it is easy to implement in the computer.

The first requirement is very important since if the search space is not limited it could happen that the genetic evolution never finds the optimal solution (not even by exhaustive search).

The fitness function is critical in our systems since it allows comparing the benefits of different proposals by the assignment of numerical values. Overcoming the difficulty that entails its development, specially having in account that the optimal solution is unknown, this function is one of the most important goals in our proposal. According to the result of the fitness function, some proposals will be "punished" and other "awarded" so the last will have more possibilities for reproduction and this way they will propagate its characteristics to get closer to the optimal solution.

Each genotype is an element in the search space and its interpretation take us to a phenotype, which is a solution proposal. Each possible solution proposal described by a phenotype must be the interpretation from at least one genotype. The difficulty in the third point is in the definition of this relation between the numerical sequence (genotypes) and the solution proposals (phenotypes).

## 2.2 Operation of a Simple GA

A simple genetic algorithm can be formally represented as follows:

```
generate initial population, G(0);
evaluate G(0);
t:=0;
repeat
   t:=t+1;
   generate G(t) using G(t-1);
   evaluate G(t);
until find a solution;
```

First, the initial population is created, constructed by a set of character strings that represent solution proposals for the problem. Then, each individual is evaluated with the fitness function to know its benefits. By means of the aptitude of each individual, a selection must be made for combining and this way produce next generation (presumably, the "best ones" will be selected). This sequence can be iterated until the best of the found solutions achieves our expectations (Koza, 1995).

The generation of individuals for next generation involves the combination phase, this is, the copy of portions of code from selected parents, and mutation, this is, the random introduction of changes in code of individuals of new generation (Srinivas, 1994).

# 3 GA FOR THE DESIGN OF CRYPTOGRAPHIC PROTOCOLS

## 3.1 Cryptographic Protocols and its Evaluation

A protocol is a set of rules or conventions that define a message interchange among two or more parts. Security protocols are designed so that the parts safely communicate over an insecure network. Security requirements include, among others, secrecy, authenticity, integrity and no-repudiation.

In cryptographic protocols, all or some messages can contain encrypted fields.

A cryptographic protocol evaluation must return a value that indicates its "goodness". This goodness can be determined considering characteristics such as if a party does not receive or understand a message intended for him, or if a message is received redundantly, or if the receiver can't demonstrate the sender knows the message.

Depending on the application field of the evaluated protocol, some additional parameters or requirements can be defined for being satisfied.

## 3.2 Previous Works

Automatic evaluation of security protocols and the metaheuristics application for the search of security protocols has been studied previously by different authors.

Ocenásek Pavel, PhD student from the Czech Republic, presents in 2005 an article as part of his works on his doctoral thesis proposing the use of GA for the design of security protocols (Pavel, 2005).

Chen Hao, John Clark and Jeremy Jacob presented an article in 2004, which explains the development of an automatic system for designing security protocols. They don't use GA, but another metaheuristic technique called "simulated annealing", on which some physical principles related with materials exposed to changes of temperature are simulated (Hao, 2004). The proposed tool is based in the BAN logic (Burrows, Abadi and Needham) that introduces an inference rules set that focus on the evolution of beliefs of honest parties involved in a protocol as they exchange information (Burrows, 1990).

Brackin present a high level language for applying formal methods to the security analysis of cryptographic protocols. This language is applied in a tool called CAPSL, developed by the United States Navy (Brackin, 1999).

# 4 TOOL FOR THE DESIGN OF CRYPTOGRAPHIC PROTOCOLS USING GA

## 4.1 Operational Bases

In the tool for the design of cryptographic protocols, the genetic algorithms were programmed as follows. The selection is by rank while applying elitist selection copying the two best individuals from each generation to next. Mutations are introduced to individuals and to genes, and only one helix is used. Next parameters can be adjusted: Number of helix, symbols, genotype and population size, weight of evaluation aspects, mutation and permutation probabilities, and number of crossover points.

## 4.2 Software Tool

The proposed tool was built, in its last version, to solve problems that require the use of public and symmetric keys, with or without a certifying authority. At this preliminary stage, elements like *hash* function, *timestamps* and *nonces* are not used. Neither is considered *man in the middle* attacks, or the introduction of data from previous runs of protocols. Of course, those aspects will be considered for future versions of the system.

Since each individual from population is a protocol to evaluate, individual evaluation is protocol evaluation. First generation protocols are randomly defined, so their results will show very low values.

Next generation's individuals will be the result from combinations of best individuals from previous generations, so their characteristics will be better.

The proposed tool works with populations of individuals. The first population, this is, the first generation is built randomly. Each individual is defined by a genetic code, a series of N helix, this is, N series of numbers. Each number can take a value from zero to a specified maximum number. Numbers from individuals of first generation are random.

For evaluate each individual, its genetic code is interpreted according a criteria that considers the combination of the numeric values stored in the helix, from where a series of binary values are extracted. This series is subdivided to establish the possible values for parameters that define the phenotype from the individual. These parameters are the characteristics that will be evaluated from each protocol.

The most complex process in evolution is the evaluation, because this will have to show with numbers the goodness of the analyzed protocols. This function is built so it must test step by step the operation of security protocols, registering with detail the consequences.

The phenotype from the individual is obtained, by getting the binary string from the helix combination on the individual's genotype, and subdividing the bits for obtaining the parameters of messages defining the protocol.

Non valid messages, like having the same origin and destiny or not having any information, are eliminated. Non valid keys, like being redundant or data not suitable for be a key, are eliminated too. Non-existent key references are not eliminated because they could be existent later.

Several cyclical loops are initiated to check each message, for each entity, trying to solve each key and register new data if there is any. All entities try all messages because it is considered that all entities could be listening all communications. New data acquired by entities are registered indicating if it was directed to him, and if it was authenticated. These operations are repeated until there is not possibility of decoding any messages by any entity.

The value to be assigned to the protocol is calculated according next criteria: number of achieved goals, redundant received data, times that data was received before a required key, data leaked, number of messages sent, cost by sending long or short messages, and cost for encoding with public or symmetric keys.

In the program, the user can select among a list of problems to solve, which are described in a external file that can be edited. There are sections for adjusting GA parameters (population size, maximum number of messages for protocol and mutation probabilities, and weight parameters for evaluation of protocols (goal achieved, pain for redundancy, late decrypting, leakage, number or messages, cost for transmitting or encrypting large or small messages). There are controls for start, stop or restart the advance of generations. It is shown on screen the best evaluated protocol of current generation and it can be written to a file.

# 5 TOOL VALIDATION

The intention on this project is to generate new protocols to solve security problems, but at this point it has been validated with known protocols generation, for simple problems. These problems were used for adjusting evaluation criteria, needed for finding a solution. The tool proposed simple protocols for problems on data exchange considering aspects that the tool evaluates, which are: public and symmetric keys, certifying authority, and the previously described evaluation criteria.

## 5.1 Example

A and B rely on Authority C to know public keys, but they can use symmetric keys for data exchange.

```
Mejor protocolo:
 Mensaje 1  C -> A: {  Kb  }Kc-1
 Mensaje 2  C -> B: {  Ka  }Kc-1
 Mensaje 3  A -> B: { { Kab }Kb }Ka-1
 Mensaje 4  A -> B: {  Xa  }Kab
 Mensaje 5  B -> A: {  Xb  }Kab
Datos adquiridos:
A: Kb Xb
B: Ka Kab Xa
C:
D:
```

Because of the cost for encrypt large messages with public keys, the system evaluates better to protocols that exchange data with symmetric keys.

# 6 CONCLUSIONS AND FUTURE WORK

In this article we have presented a software tool that applies the metaheuristic techniques of GA for the creation of cryptographic protocols.

The first versions of the tool were for a concept proof. They were encouraging and motivated the addition of new elements and refining of parameters for the evaluation of protocols for more complex problems. All parameters required fine adjust to be able to solve all problems without having to adjust them. They needed to be further adjusted to reflect real life costs.

There are a lot of pending aspects that will be added into the tool in a short term: nonces and hash functions.

In a middle term there will be considered the generation of attacks for a testing protocol, using genetic algorithms.

In the long term it will be considered the possibility for the generation of security protocols for electronic voting and auctions.

# ACKNOWLEDGEMENTS

This work has been partially supported by the research project SECONNET (TSI2005-07293-C02-01).

# REFERENCES

Holland, J 1975, *Adaptation in natural and artificial systems*, University of Michigan Press, Michigan.

Koza, J 1995, 'Survey of genetic algorithms and genetic programming', *Proceedings of the WESCON'95*, pp. 589-594

Srinivas, M, & Patnaik, L 1994, 'Genetic algorithms : a survey', IEEE Computer, pp. 17-26.

Pavel, O 2005, 'The security protocol design using genetic algorithms', *Preceedings in the 11th Conference and Competition STUDENT EEICT*, vol., no., pp. 576-580.

Hao, C, Clark, J & Jacob J 2004, 'Automated design of security protocols'. *Computational Inteligence*, vol. 20, no. 3.

Burrows, M, Abadi, M & Needham, R 1990, 'A logic of authentication', *ACM Transactions on Computer Systems*, vol. 8, no. 1, pp. 18-36.

Brackin, S, Meadows, C & Millen, J 1999, 'CAPSL interface for the NRL protocol analyzer', *Proceedings of ASSETT99*.