Nature–Inspired Synthesis of Rational Protocols

Almudena Alcaide, Juan M.E. Tapiador,
Julio C. Hernandez-Castro, and Arturo Ribagorda

Computer Science Department, Carlos III University of Madrid
Avda. Universidad 30, 28911, Leganes, Madrid
{aalcaide,jestevez,jcesar,arturo}@inf.uc3m.es

Abstract. Rational cryptography is an emerging field which combines aspects traditionally related to security with concepts described in economic theoretical frameworks. For example, it applies game theory concepts to address security problems arising when executing cryptographic protocols. The aim is to replace the assumption of a worst–case attacker by the notion of rational agents that try to maximize their payoffs. In this work, we define a formal framework and a meta–heuristic technique for the automated synthesis of multi–party rational exchange security (M–RES) protocols. We provide experimental results for a simple scenario where a 3–party rational exchange protocol is automatically designed.

1 Introduction and Motivation

The *exchange problem* of how to design a general procedure according to which several parties can exchange items in a *fair* manner has attracted much attention throughout the years. Interest in this class of protocols stems from its importance in many applications where disputes among parties can occur, such as digital contract signing, certified e–mail, exchange of digital goods and payments, etc. Roughly, the property of fairness means that no party should reach the end of the protocol in a disadvantageous position. Still, there exist no protocol according to which a number of parties can exchange items in a fair manner exclusively by themselves, assuming that misbehaving parties take part in the protocol ([1].) As a result, all fair exchange protocols require a trusted third party (TTP) in order to preserve fairness during the exchange.

By contrast, *rational* exchange protocols do have the enormous advantage of not needing a TTP. Informally, a rational exchange protocol cannot provide fairness, but it ensures that rational (i.e. self–interested) parties would have no reason to deviate from the protocol, as misbehaving does not result beneficial. The work presented in this paper focuses on the automated design of this type of protocol (rational exchange protocols) in multi–party environments.

Next we motivate the need for this approach and introduce some related work.

Shortage of rational proposals. As it is only recently that rational exchange schemes have been considered as an alternative solution to the exchange problem,

G. Rudolph et al. (Eds.): PPSN X, LNCS 5199, pp. 981–990, 2008.

there are very few rational exchange protocols proposed in the literature (see e.g. [2], [3].)

Automated analytical tools versus automated designing tools. With regard to the formal analysis of security protocols, several automated tools have been presented over the years (see [4] for an excellent survey.) In every case the focus has always been on the validation of existing schemes. We intend to adopt a relatively novel approach integrating formal verification within the designing methodology.

Meta–heuristic search for automated protocol synthesis. It is clear that the number of possible protocols achieving a set of goals from a set of initial assumptions grows exponentially as the number of goals or the number of participant entities rise. In this context, a designing methodology based on meta–heuristic search appears to be a reasonable option. Hao, Clark and Jacob were the first in applying these techniques for the synthesis of protocols that are provable correct and satisfy certain security criteria ([5] and [6].) In their work, they present an automated tool, based on Simulated Annealing, which finds security protocols that achieve certain goals from a set of initial assumptions. In later work [7] the authors apply a different heuristic technique based on genetic algorithms for the synthesis of provably secure protocols. A similar approach is also adopted by Park et al. in [8] to the synthesis of cryptographic protocols for a fault–tolerant agent replication system.

1.1 Overview of Our Work

In this paper, we describe a framework for the automated synthesis of rational exchange protocols (Section 2). The practical implementation of this formalism results on an application designed to find multi–party rational exchange security (M–RES) protocols for specified scenarios. We will illustrate its practical application within a 3–party scenario (Section 3) where our heuristic technique finds rational solutions very efficiently.

2 Foundations

Simple linear structures such as vectors and matrices will be used to represent all aspects of a multi–party exchange problem. Prior to describing the model in detail, the following definition will serve to unify notation throughout the paper:

Definition 1 (Exchange protocol). *Given a set of entities $\mathcal{P} = \{P_1, \ldots, P_v\}$ and a set of items $\mathcal{O} = \{o_1, \ldots, o_m\}$, an exchange protocol Π consists of n steps, each denoted by* $(t)\ P_i \rightarrow P_j : o_{t_1}, \ldots, o_{t_{k_t}}$*, where:*

- *$t = 1, \ldots, n$ is the step number,*
- *P_i, $P_j \in P$, $i \neq j$, are the sender and receiver of the message, respectively.*
- *$\{o_{t_1}, \ldots, o_{t_{k_t}}\} \subseteq \mathcal{O}$ are the items P_i sends to P_j, subject to P_i owning those items at step t of the protocol.*

Note that this definition merely describes a series of messages being exchanged between participants so that, at the end of the protocol execution some entities would have lost control over some of their items as well as having gained access over new ones. Further along in the synthesis process, a fitness function will decide how good a protocol is in giving solution to a specific exchange problem.

2.1 Protocol Representation

Protocols described in Definition 1 will be represented by a series of matrices.

Protocol Matrix. A protocol Π is represented by a matrix $S^{\Pi} \in \mathcal{M}_{n\times(m+2)} = [s^{\Pi}_{i,j}]$ of integers, where each row is interpreted as a message in which the first two components identify the sender and the receiver of the message, respectively, and the rest of the row components represent the items being sent.

Although matrix S^{Π} represents the series of steps that participant entities have to take along a protocol execution, the actual real message content being sent at each step is subject to: (1) the sender entity holding the referred items at that point in the protocol run; and (2) those items being *accessible* to that sender. Different situations could derive in a *non–accessible* status of an item o_j for a particular entity P_i. For example, if an item o_j is encrypted and entity P_i does not hold the decryption key. Something similar happens if entity P_i is able to generate item o_j but it needs to gain access to other items in order to do so. In this case, item o_j must remain non–accessible until gaining control over the rest of the required tokens. During the protocol execution this kind of information, which is specific to the particular exchange problem at hand, will be captured in two additional matrices: a matrix $H(t)$ denoted *state matrix* and a matrix R, denoted *inter–dependency matrix* describing items' dependency relations. Both structures are described below.

State Matrix. For each step t in the protocol $t = 1, \ldots, n$, matrix $H(t) = [h_{i,j}(t)] \in \mathcal{M}_{v\times m}$ will capture the possessions of each party at the end of such a step. At the initial step $(t = 0)$ the matrix will represent the possessions of each different entity prior to the exchange.

Inter-Dependency Matrix. A matrix $R = [r_{i,j}] \in \mathcal{M}_{(v\times m)\times(v\times m)}$ will capture the inter–dependency relations for each $h_{i,j} \in H$ for a given exchange problem. Two different types of dependency relations, *positive* and *negative*, can be expressed within the model as follows:

- Items o_i and o_j are positively related if when o_j is non–accessible then, gaining access to o_i implies gaining access to item o_j too.
- Items o_i and o_j are negatively related if when o_j is non–accessible, then receiving o_i implies making item o_i non–accessible.

Further and more complex dependency links may be represented in matrix R, involving several items. The only restriction imposed by this representation is that negative and positive relations between any two given elements cannot be simultaneously expressed.

Updating the State Matrix. Initially, a candidate solution consists of a protocol matrix S^{Π}, a state matrix $H(0)$, and an inter–dependency relation matrix R, specific to the exchange environment. As the protocol execution progresses, the state matrix H is updated according to the instructions given in the protocol and the positive and negative restrictions imposed by matrix R. At the end of the protocol execution $H(n)$ will reflect the possessions that each entity holds and also those items that each entity has lost control over. How good the protocol S^{Π} is in giving solution to a particular exchange problem will be decided by a fitness function.

2.2 Fitness Function

A fitness function is individually defined for each participant of the protocol to evaluate the gains obtained at each step of the scheme. The search will aim at finding exchanging schemes which maximize each individual fitness function.

Benefit Matrix. In our model, all participants assign every item involved in the exchange a particular value. Those values serve to represent each individual's set of requirements and are captured in matrix $B = [b_{i,j}] \in \mathcal{M}_{v \times m}$, denoted *benefit matrix*. A more formal description of this matrix is given next:

$$b_{i,j} = \begin{cases} 1 & \text{iff } P_i \text{ incurs cost when losing control over } o_j \\ 0 & \text{iff item } o_j \text{ is of no value to participant } P_i \\ -1 & \text{iff } P_i \text{ obtains benefit when losing control over } o_j \\ & (\textit{Via coalitions or incentives}) \\ > 1 & \text{iff item } o_j \text{ is required by participant } P_i \\ & \textit{(It represents the value that item } o_j \textit{ is worth to entity } P_i, \\ & \textit{if and only if, } o_j \textit{ becomes accessible to } P_i\textit{)} \end{cases} \tag{1}$$

Maximum and Minimum Benefit Values. The following criteria will serve to: (1) compute the maximum benefit that an entity can obtain in a single protocol run; and (2) compute the minimum benefit that each entity P_i will obtain, which satisfies its requirements.

- A maximum benefit value $\hat{b}_i$ represents the payoff obtained when the outcome of the protocol run is the most favorable for entity P_i. It is computed considering that the entity has gained access to all the required items, it has sent all items for which losing control over is beneficial and has kept all items for which sending represents a cost.
- Minimum benefit value $\bar{b}_i$ represents the minimum payoff that entity P_i would expect to obtain with the exchange. The minimum that a rational entity will consider as a satisfactory exchange is that in which the entity has gained access to all required items, has had to lose control over items for which sending represents a cost, and it does not possess any of the items for which relaying is beneficial.

We will now define a fitness function to compute participant "fitness" (i.e. benefit attained) after each step in the protocol, as well as global protocol fitness at the end of a protocol run.

Utilities. At each step t of the protocol (after updating state matrix $H(t)$ according to the transference of items), we shall refer to the gains achieved by a player so far as "utility" or "payoff" values. Each P_i's current utility at step t, $(0 \leq t \leq n)$ can be denoted as $u_i(t)$.

Differential Utilities. The *differential utility* du_i for a player P_i between steps t_1 and t_2, with $0 \leq t_1 \leq t_2 \leq n$, is defined as:

$$du_i(t_1, t_2) = u_i(t_2) - u_i(t_1) \tag{2}$$

Additionally, a *global differential utility* dU will measure the overall fitness of a protocol solution S^{Π}. This can be defined as the sum of the benefit attained by each participant at the end of the execution:

$$dU(S^{\Pi}) = \sum_{i=1}^{v} du_i(0, n) \tag{3}$$

2.3 On the Solution Space

Given the formalism just described, our goal will be to explore the space of all exchange protocols to find schemes which are:

1. **Feasible.** That is, the exchange described by the protocol solution S^{Π} would represent a feasible transference of the required items between each protocol participant and,
2. **Rational.** During a protocol execution, there may be steps at which players run into a temporarily "worse" state (i.e. $du_i(t, t+1) \leq 0$.) However, the relevant factors which ensure rationality of the scheme are:
   i. At the end of the protocol run, P_i must have gained enough differential utility. If $du_i(0, n) > 0$, the exchange is profitable to P_i, if $du_i(0, n) < 0$, the exchange is non–profitable to P_i and, when $du_i(0, n) = 0$ indicates that the exchange is of no use to P_i.
   ii. For each participant, the utility $u_i(n)$ must satisfy the minimum required by P_i $(u_i(n) \geq \bar{b}_i)$.
   iii. Finally, entities having attained their required minimum $\bar{b}_i$ in an intermediate step, should not be considered as active participants for the rest of the protocol. That is, entities achieving their goals must be forced to quit the protocol execution.

How Many Exchange Protocols Exist? As described in Section 2.1, a protocol is represented by a matrix $S^{\Pi} \in \mathcal{M}_{n \times (m+2)}$. An estimate of the total number of possible exchange protocols can then be given by:

$$\mathcal{O}\big(\frac{v!}{(v-2)!}2^{nm}\big) = \mathcal{O}\big(v(v-1)2^{nm}\big) = \mathcal{O}(v^2 2^{nm}) \tag{4}$$

where n is the number of protocol steps, v is the number of entities and m is the number of tokens involved in the exchange.

It is difficult to determine how many of these protocols represent *feasible* solutions to a specific exchange problem. Even more challenging is to estimate how many of those feasible solutions represent a *rational* exchange. An heuristic search based on Simulated Annealing will assist in finding those protocol designs within the solution space of a given multi–party exchange problem, which satisfy the above conditions of feasibility and rationality.

3 Automated Synthesis of a 3–RES Protocol

For the purpose of this paper, we will focus on a particular 3–entity exchange problem. For every participant entity we will provide a series of initial assumptions and goals which will be represented using the matrices described in Section 2.

3.1 A 3–RES Problem

Initial assumptions and other aspects of the particular exchanging problem are formalized as follows:

1. The specific exchange problem will consist of an entity P_0 which aims to collect a series of electronic items from entities P_1 and P_2, delivering the appropriate tokens in return. All entities, P_0, P_1 and P_2, are considered to be rational (aimed to maximize their payoffs). The following items are involved in the scheme:
   - o_0: Request token issued by P_0 containing a description of the item that P_0 requires from P_1.
   - o_1: Request token issued by P_0 containing a description of the item that P_0 requires from P_2.
   - o_2: Return token issued by P_0 for P_1 in return for o_4.
   - o_3: Return token issued by P_0 for P_2 in return for o_5.
   - o_4: Customized item issued by entity P_1 as specified by P_0 in o_0.
   - o_5: Customized item issued by entity P_2 as specified by P_0 in o_1.
2. None of the collected items in isolation is of any value to entity P_0. In other words, P_0 is interested in collecting all (i.e. o_4 and o_5) or none of these items.
3. Entities must choose an arbitrary positive integer greater than one, for each one of their required items. These values will represent the payoff associated to gaining access to such items.
4. Finally, the nature of these items is such that their utilities only become available when the corresponding tokens are delivered in return. Although this restriction seems hard, there are a few real life examples where items are of this nature. For example, P_0 could be a user trying to book a holiday package consisting of accommodation, flights and tickets for a local tourist attraction. User P_0 needs either all or none of the required items and, additionally, no item becomes available unless the providers of the required services have received payment.

3.2 Search Technique and Parameterization

Simulated Annealing (SA) [9] will be used as search technique. The basic algorithm has been slightly modified to stop when the first rational exchange protocol which satisfies the requirements is found. (This can be done by previously computing the minimum required global fitness.)

A simple random mutation mechanism is employed as move operator. Given a candidate solution (specified by a protocol matrix S^{Π}), a neighbor is obtained by randomly modifying a percentage of its elements. We will refer to the amount of elements mutated in the matrix as the *moving rate.* The different moving rates considered in the experimental work are: 1%, 5%, 10%, 20%, 30%, 40%, 50%, 60%, 70% and 80%.

The acceptance criterion in SA is given by:

$$s' \text{ is accepted if } f(s') - f(s) > T_i \ln u \tag{5}$$

where s and s' are, respectively, the current and mutated solutions, T_i is the current temperature, and u is a random number uniformly generated in $[0, 1]$.

At each cycle, the temperature is geometrically decreased by:

$$T_{i+1} = \alpha T_i \tag{6}$$

$0 < \alpha < 1$ being the cooling factor.

Note that, after m cycles the temperature is $T_m = \alpha^m T_0$ where T_0 is the initial temperature. For T_m to be very close to 0 (say $\epsilon = 10^{-6}$) after m cycles, a cooling rate of:

$$\alpha = \left(\frac{\epsilon}{T_0}\right)^{\frac{1}{m}} \tag{7}$$

is needed.

For our experimental work, these SA parameters have been adjusted according to the definition of two different profiles. Both, **profiles (I) and (II)**, will satisfy the following property: in the first cycle, the probability of accepting a bad move which decreases the global fitness value by just one unit will be approximately 0.5. Moreover, in profile (I), by half the total number of cycles, the probability of accepting a bad move which decreases the global fitness value by more than one unit will be almost 0. So from exactly half the total number of cycles onwards, the search will behave as a pure hill climbing (HC.) By contrast, in profile (II), the probability of accepting a bad move which decreases the global fitness value by more than one unit will be almost 0 by one quarter of the total number of cycles. In this case, from exactly one forth of the total number of cycles onwards, the search will behave as a pure hill climbing.

Other parameters for our specific problem are the following. There are 3 parties in the exchange which exchange 6 items according to the scenario described in Section 3.1. The maximum allowed number of messages per protocol is set to 10, with each message consisting of at most 6 items. Note that with these parameters the search space (possible number of protocols) is $\mathcal{O}(2^{63})$, according to expression (4).

Table 1. Rate of success (RS) and average number of protocols evaluated (NPE) per trial and moving rate (MR). Results estimated over 500 trials.

	Hill Climbing		**SA (Profile I)**		**SA (Profile II)**	
MR	RS	Avg. NPE	RS	Avg. NPE	RS	Avg. NPE
0.01	64.8%	108348	79.4%	82434	71.6%	90830
0.05	93.6%	47068	99.6%	27637	97.2%	29765
0.1	97.6%	36464	99.6%	24833	99.4%	23879
0.2	98.2%	35274	99.2%	32709	99%	29981
0.3	90.4%	59144	98.4%	47924	95%	50588
0.4	56.6%	42754	75.8%	67340	68.4%	55958
0.5	18.6%	26387	30.20%	56718	23.8%	44678
0.6	5%	14311	11.2%	40484	7%	29673
0.7	1.4%	11281	5%	37466	3.2%	24909
0.8	2%	10047	3.6%	34830	1%	24535

(a)

Random Search	
RS	Avg. NPE
2.0%	19823
6.4%	96975
12.4%	186519
23.0%	355611

(b)

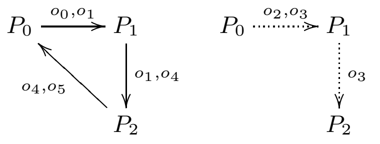
3.3 Results

Extensive experimentation has demonstrated that around 200 cycles with 1000 moves in the SA inner loop are sufficient to reach solutions in reasonable time (around 1 or 2 minutes.)

Table 1(a) shows the results obtained for the two SA profiles and different moving rates (column MR). The rate of success (column RS) represents the percentage of executions attaining a feasible rational protocol over 500 trials. The average number of protocols evaluated is indicated in column Avg. NPE. Finally, both SA profiles (I and II) are compared with the results obtained when applying a classic Hill Climbing algorithm (HC.)

In both SA profiles, the best results are obtained with a moving rate of 0.1, which results in around a 99.5% of success (i.e. almost every execution produces a valid solution) by evaluating approximately 24500 protocols. These numbers imply synthesizing a protocol for this scenario in less than 1 minute in a common laptop. The success rate for slightly lower or higher mutation rates are similar, though the number of total candidates evaluated before reaching a solution grows considerably, thus resulting in a more inefficient search. As expected, higher mutation rates transforms the search in an almost random procedure with fewer chances to succeed. Further comparatives are shown in Table 1(b) where a random search is applied to resolve the same problem.

All in all, the best rates of success are systematically achieved by SA. Even though a simple HC technique attains very good solutions too, the average number of protocols evaluated per trial serves as an experimental proof of efficiency in favor of a more sophisticated heuristic based on SA. Furthermore, our preliminary experimentation indicates that this is certainly the case in more complex exchange scenarios. Finally, as for a pure random search, the numbers are several orders of magnitude below the results obtained by any of the other two techniques.



(a)

$(1)\ P_0 \rightarrow P_1 : \{o_0, o_1\}_{K_{P_1}}$
$(2)\ P_1 \rightarrow P_2 : \{\{o_1, o_4\}_{K_{P_1}^{-1}}\}_{K_{P_2}}$
$(3)\ P_2 \rightarrow P_0 : \{o_4, o_5\}_{K_{P_2}^{-1}}$
$(4)\ P_0 \rightarrow P_1 : \{o_2, o_3\}_{K_{P_1}}$
$(5)\ P_1 \rightarrow P_2 : \{\{o_3\}_{K_{P_1}^{-1}}\}_{K_{P_2}}$

(b)

Fig. 1. A synthesized 3–entity rational exchange protocol. The protocol runs in the two phases illustrated on the right.

3.4 An Example of 3–RES Protocol

Figure 1(a) shows an example of a synthesized protocol for the problem described in Section 3.1. Further security refinements are applied to each message resulting in the scheme shown in Figure 1(b) (K_{P_i} and $K_{P_i}^{-1}$ denote P_i's public and private keys, respectively.)

The 3–RES protocol synthesized using our proposed approach can be formally proven rational using techniques based on game theory and backward induction (see [10].) Informally, here are some aspects of the formalism which ensure that the scheme is a feasible rational solution satisfying all participants' sets of requirements:

- **From entity's P_0 point of view.** As stated in the initial assumptions, items o_4 and o_5 are of no use to entity P_0 until the corresponding return items o_2 and o_3 have reached entities P_1 and P_2, respectively. To this regard, and since entity P_0 requires either all or none of these items, entity P_0 is *rationally* forced to perform step (4) of the protocol.
- **From entities P_1 and P_2 point of view.** Again, the assumption of P_0 requiring either all or none of the items forces (rationally) entity P_1 to send messages (2) and (5) and entity P_2 to send message (3).

Therefore, no entity would unilaterally deviate from the 3–RES protocol as they could not obtain better utility values in doing so. The scheme is then a rational solution.

4 Conclusions

Traditionally, automated tools have always been applied to the analysis and verification of existing security protocols. In this paper we have adopted a new approach, ensuring rationality as part of the automated design of an exchange scheme.

For the purposes of this work, we have designed and implemented a 3–RES search algorithm based on Simulated Annealing. Moreover, the formal foundations of our methodology ensure high levels of flexibility and scalability for any multi–party rational exchange problem.

References

1. Pagnia, H., Gärtner, F.: On the impossibility of fair exchange without a trusted third party. Technical report, Darmstadt University of Technology, Department of Computer Science (1999)
2. Buttyán, L.: Building blocks for secure services: Authenticated key transport and rational exchange protocols. Technical report, Swiss Federal Institute of Technology. Lausanne (EPFL), Ph.D. Thesis No. 2511 (2001)
3. Syverson, P.: Weakly secret bit commitment: Applications to lotteries and fair exchange. In: Proceedings of the 11th IEEE Computer Security Foundations Workshop, pp. 2–13 (1998)
4. Kremer, S.: Formal analysis of optimistic fair exchange protocols. Technical report, Université Libre de Bruxelles. Faculté de Sciences Ph.D. Thesis (2003)
5. Clark, J., Jacob, J.: Protocols are programs too: the meta-heuristic search for security protocols. Information and Software Technology 43, 891–904 (2001)
6. Chen, H., Clark, J., Jacob, J.: Automatic design of security protocols. Computational Intelligence 20, 503–516 (2004); Special Issue on Evolutionary Computing in Cryptography and Security
7. Clark, J., Jacob, J.: Searching for a solution: engineering tradeoffs and the evolution of provably secure protocols. In: Proceedings IEEE Symposium on Security and Privacy (2002)
8. Park, K., Hong, C.: Cryptographic protocol design concept with genetic algorithms. In: KES (2), pp. 483–489 (2005)
9. Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by simulated annealing. Science 220, 671–680 (1983)
10. Alcaide, A., Estévez-Tapiador, J., Hernandez Castro, J., Ribagorda, A.: A multiparty rational exchange protocol. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2007, Part I. LNCS, vol. 4805, pp. 42–43. Springer, Heidelberg (2007)