Mobile Agents: Preserving Privacy and Anonymity

Aneta Zwierko[1] and Zbigniew Kotulski[1,2]

[1] Warsaw University of Technology, Faculty of Electronics and Information Technology, Institute of Telecommunications
azwierko@tele.pw.edu.pl

[2] Polish Academy of Sciences, Institute of Fundamental Technological Research
zkotulsk@ippt.gov.pl

Abstract. The mobile agent systems have been well known for years, but recent developments in the mobile technology (mobile phones, middleware) and the artificial intelligence created new research directions. Currently being widely used for the e-commerce and network management are entering into more personal areas of our life, e.g., booking airline tickets, doing shopping, making an appointment at the dentist. Future agents are becoming more like our representatives in the Internet than simple software. To operate efficiently in their new role they need to have the same capabilities as we do, showing their credentials when required and being anonymous when needed. Still they have to fulfill all security requirements for agent systems, including confidentiality, integrity, accountability, and availability. This paper focuses on providing mobile agents with anonymity and privacy. The proposed schemes are based on different cryptographic primitives: the secret sharing scheme and the zero-knowledge proof. The paper also includes a discussion of security of the proposed schemes.

1 Introduction

A software agent is a program that can exercise an individual's or organisation's authority, work autonomously toward a goal, and meet and interact with other agents ([10]). Agents can interact with each other to negotiate contracts and services, participate in auctions or barter. Agents are commonly divided into two types

- stationary agents,
- mobile agents.

The stationary agent resides at a single platform (host), the mobile one can move among different platforms (hosts) at different times.

Agent systems are used for intrusion detection; combined with meta-learning agents create even more powerful tools for detecting security threats in the network environment ([6]). Other fields where agent systems are widely used are management systems for telecommunication networks. The most popular telecommunication management protocol, SNMP (Simple Network Management Protocol), existing for over 20 years, is based on the idea of an agent and a manager. Many similar systems have been proposed in the past and still exist. Mobile agents are also well suited for software distribution and can provide adaptive responses to network events.

Another practical application field for agents systems is the e-commerce where mobile agent-based applications have been proposed and are being developed for a number of diverse business areas like: contract negotiations, service brokering, auctions, and stock trading ([13]). For example, manufacturers can negotiate the delivery of goods and services with suppliers utilising agents. The agents may need to access the supplier's database, transfer money, and negotiate terms of delivery, warranties, and service contracts. Mobile agents representing bidders may meet on an auction house's platform to engage in blind, straight, or Dutch auctions, each employing different strategies and having different financial constraints.

Also producers of the small mobile equipment, such as: cellular phones, personal organisers, car radios, and other consumer electronic devices are introducing more and more functionality into their products, becoming the focus of agent developers. These devices are not continuously on-line and can greatly benefit from a mobile agent's ability to operate autonomously. Agent developers often cite the example of a user launching an agent to make travel, hotel, and dinner reservations by negotiating with other agents, as an illustrative scenario for mobile agent technology.

One of new directions for the development of the agents' systems is a communicative intelligence. The future agents' systems are not only supposed to be communicating or/and interacting with each other but also with real people. They should have the same capabilities as people: to hide their identity when convenient and show their credentials when needed. Preserving privacy and anonymity, so easy in the human environment, is one of the current most significant problems in the agent systems.

Many agent systems are designed to interact with people. Already the research is going on toward agents that will act accordingly to different customs and depending on geographical area where they work; we can imagine an agent being e-teacher, who will differently communicate and work with small child, a school student and an adult. In many other cases such agents, equipped also with some kind of intelligence, will need anonymity, e.g., when working as brokers or insurance agents.

Also many e-commerce systems treat so called trade agents as users' representatives. They can gather and analyse information using artificial intelligence methods (e.g., expert systems), negotiate on behalf of the user and present a set or a subset of negotiation results.

Another research direction is societies of artificial agents and social agents. Social agents are entities that have their own goal and principles. They can interact with one another and exist in a social context. Several types of social agents can be developed: the simplest would be a reactive agent. It just receives a signal from the environment and reacts to it. It has no memory of the past and no goal for the future. The most sophisticated would be the anticipatory agent. It not only has a memory of its past but also has some predictions about the future. It makes decisions basing on these anticipations. Such agents are based on cognitive models of different types.

Such an understanding of the agent systems will widespread with growing use and development of mobile equipment. Also the need for more sophisticated services and systems will be more urgent.

2 Related Work

Many different methods for securing agent systems exist. Here some most popular and interesting are discussed. Note that almost none provide anonymity. However, the systems providing anonymity to different services exist and also are presented in this section.

The most popular methods of securing agent systems are policies. An example of such a concept is the allocation of privileges ([9]). It utilises different types of certificates: the attribute and the policy certificates. To create them an additional Public Key Infrastructure (PKI) is needed. The attribute certificate is bound with an agent: it protects its security relevant information from alteration and assigns privileges to an agent. The second type of certificate is associated with a host and it contains specific policy rules for a given host. This solution does not provide any anonymity or privacy to the agents. Its security is based on the PKI.

Another approach to the agent's security is proposed in [4]. The idea is based on the one-round secure computation and is somehow similar to the concept of the computation with encryption functions, one of the most important methods for protecting agent's integrity. The scheme enables an agent to hide the data computed on one agent platform from all other hosts. The secure computation protocol enables the agent to compute the required data in a way that host cannot learn anything about input, only the output. However, this system does not provide any authorisation or authentication and requires additional communication between the hosts.

In [1] mobile system based on domain architecture is proposed. The system contains centralised authorisation servers, authorisation tokens and authorisation agents. The security of the system is based on PKI. The agents are used to provide authorisation data for any mobile unit, e.g., other agents or mobile hosts.

Another kind of a security scheme for an agent system is evaluated in [11]. It is based on a concept of a master agent (stationary) and slave agents (mobile). The PKI is used to provide security and the system does not offer anonymity to users.

A different proposal of securing an agent system (large-scale and distributed) is outlined in [19]. Its core is the SPKI/SDSI chains of trust and it utilises the certificate delegation infrastructure to provide decentralised authorisation and authentication control. Also the idea of a federation of hosts and a mutual authentication of agent's platforms is used. No anonymity in this system is provided.

Also many similar systems, mostly based on PKI exist; see [3] and [7].

A scheme preserving anonymity is proposed in [5]. The scheme is based on a credential system and offers optional anonymity revocation. Its main idea is based on the oblivious protocols, the encryption circuits and the RSA assumption. However, the possible applications for agent systems are not presented.

Another approach to hide the senders and receivers of messages is presented in [2]. The basic idea was inspired by a public transportation system that naturally hides communication patterns. The "buses" represent messages and each piece of information has its "seat". The buses travel specific, initially chosen routes. Different deterministic and randomised protocols with possible improvements are presented and analysed. However, a possible utilisation for the agent system is not proposed.

One of the systems providing the anonymity during browsing WWW is *Crowds* ([17]). The system hides the action of one user among actions of other users. All users are called *the crowd* and the server issues the request on behalf of its users. The end servers cannot determine which user in reality performed some action.

Some other system providing anonymity, VAST, was described in [14]. Many other anonymity systems were proposed, mostly without any application to agent systems.

3 Security

Providing security is complex and tough for most existing services. It is even more problematic in a distributed environment, such as agents' systems. Most important security requirements are ([10]):

- Confidentiality: any private data stored on a platform or carried by an agent must remain confidential. Mobile agents also need to keep their present location and route confidential.
- Integrity: the agent platform must protect agents from unauthorised modification of their code, state, and data and ensure that only authorised agents or processes carry out any modification of the shared data.
- Accountability: each agent on a given platform must be held accountable for its actions: must be uniquely identified, authenticated, and audited.
- Availability: every agent (local, remote) should be able to access data and services on an agent platform, which responsible to provide them.
- Anonymity: agents' actions and data should be anonymous for hosts and other agents; still accountability should be enabled.

Threats to security generally fall into three main classes: disclosure of information, denial of service, and corruption of information ([10]).

Threats in agent system can be categorized into four groups:

- an agent attacking an agent platform,
- an agent platform attacking an agent,
- an agent attacking another agent on the agent platform,
- other attacks.

The last category covers cases of an agent attacking an agent on another agent platform, and of an agent platform attacking another platform, and also more conventional attacks against the underlying operating system of the agent platform. In this paper we will focus on the threats from an agent's perspective. The possible scenarios of an agent interacting with a malicious host or a group of malicious hosts working together are discussed.

4 Anonymity

The anonymity is very complex and hard to provide in classical services, like browsing web. It is even more complex to provide anonymous agents. Many services require the anonymity to function as in the real world. Many of the e-commerce transactions should be anonymous. If someone is observing actions of an agent, this can be itself a source of a very useful knowledge, even without eavesdropping on agent's data. In many situations privacy and anonymity should be preserved ([12]).

Agents should be able to reveal (or not) their presence to other agents or hosts. For example, an agent shopping for goods and services may wish to do so in privacy. Also during auctions or an initial phase of negotiations agents may want to remain anonymous. In some situations the knowledge that a particular agent is interested in some kind of services can be an advantage for a vendor over its opponents. In addition, an agent may not want to disclose which hosts it has visited before the current one. It may need to keep not only its present location but also the route secret.

However, the anonymity is not always an advantage in agent systems. Every agent has to authenticate itself to other agents or hosts to be able to perform needed actions, e.g. when a financial transaction is to be carried out, the platform may require some form of authentication. Also authentication mechanisms provide accountability for user actions.

An agent's anonymity is also connected with possible security risk. In some cases the security policy of hosts does not accept anonymous agents, or offers different levels of privileges with different anonymity levels. The level of sensitivity of the transaction or data for which agent requests an access may require the agent to offer different degrees of authentication ([17]). Also sometimes the host may not be willing to accept agents that have been on certain platforms, e.g., outside the authority of certain approved security domains. In agent societies where reputation is valued and used as a means to establish trust, other agents through masquerade can harm an agent's reputation. It should be protected by an agent platform.

In this paper we propose two mechanisms of agent's authorisation preserving its anonymity at a certain level.

5 Proposals

This section describes a new idea for the authentication scheme for a mobile agent system that is preserving the agent's anonymity and privacy. Each agent has to authorise itself to the host to be able to perform any action (e.g., buy anything, start negotiations, ask for an offer, etc.). The agent should be anonymous: malicious hosts, even working together, should not be able, basing on an authorisation data, to identify actions performed by each agent. Still, this system should have some management capabilities and auditability: any authorised entity (e.g. manager) should be able to identify actions performed by each agent with every host. So, each pair agent-host should use some different authorisation data, which will be unique, but should not enable any host to differentiate between agents.

First, the utilised cryptographic primitives are briefly introduced: the Merkle's puzzles, concept of the zero-knowledge proofs and the secret sharing scheme. Then, the idea for the new system is presented and details of both proposed solutions are given.

5.1 Cryptographic Primitives

Merkle's Puzzles. Ralph Merkle introduced his concept of cryptographic puzzles in [15]. The goal of this method was to enable secure communication between two parties: A and B, over an insecure channel. The assumptions were that the communication channel could be eavesdropped (by any third party, called E).

Assume that A selected an encryption function F. F is kept by A in secret. A and B agree on a 2^{nd} encryption function, called G:

G(plaintext, some key) = some encrypted message.

G is publicly known. A will now create N puzzles (denoted as p_i, $0 \leq i \leq N$) in the following way:

$$p_i = G((K, X_i, F(X_i)), R_i),$$

where K is simply a publicly known constant quantity, which remains the same for all messages, X_i are selected by A at random, R_i are the "puzzle" parts, and are also selected at random from the range $(N \cdot (i-1), N \cdot i)$. After creating all puzzles, A sends all of them over an insecure channel to B (they can be observed by E). To solve each puzzle B must guess the R_i. For each message (puzzle), there are N possible values of R_i. If B tries all of them, he is bound to chance upon the right value. This will allow B to recover the message within the puzzle: the triple $(K, X_i, F(X_i))$. B will know that he has correctly decoded the message because the constant part, K, provides enough redundancy to ensure that all messages are not equally likely. Without this provision, B would have no way of knowing which decoded version was correct, for they would all be random bit strings. Once B has decoded the puzzle, he can transmit X_i in clear. $F(X_i)$ can then be used as the encryption key in further communications. B knows $F(X_i)$ because it is in the message. A knows $F(X_i)$ because A knows X_i, which B transmitted in clear, and also knows F, and so can compute $F(X_i)$. E cannot determine $F(X_i)$ because E does not know the F, and so the value of X_i tells E nothing. E's only recourse is to solve all the N puzzles until he encounters the correct puzzle that B solved. So for B it is easy to solve one chosen puzzle, but for E is computationally infeasible to solve all N puzzles.

Zero-Knowledge Proofs. A zero knowledge proof system ([16]) is a protocol, which enables one party to *prove* the possession or knowledge of a "secret"' to another party, without revealing anything about it, in the information-theoretical sense. Such protocols are also known as minimum disclosure proofs. The zero knowledge proof involves two parties: the prover who possesses a secret and wishes to convince the verifier that he indeed has a secret. As mentioned before, the proof is conducted via an interaction between the parties. At the end of the protocol the verifier should be convinced only if the prover knows the secret. If, however, the prover does not know it, the verifier will be sure of it with an overwhelming probability.

The zero-knowledge proof systems are ideal for constructing identification schemes. A direct use of a zero-knowledge proof system allows unilateral authentication of P (Peggy) by V (Victor) and require a large number of iterations, so that verifier knows with an initially assumed probability that prover knows the secret (or has the claimed identity). This can be translated into the requirement that the probability of false acceptance be 2^{-t}, where t is the number of iterations. A zero knowledge identification protocol reveals no information about the secret held by the prover under some reasonable computational assumptions.

Secret Sharing Scheme. A (t, n) threshold secret sharing scheme ([16]) distributes a secret among n participants in such a way that any t of them can recreate the secret. But any $t - 1$ or fewer members gain no information about it. The piece held by a single participant is called a *share* or *shadow* of the secret. A trusted authority, called a dealer, sets up a secret sharing scheme, computes all shares and distributes them to participants via secure channels. The participants hold their shares until some of them decide to combine their shares and recreate the secret. The recovery of the secret is done by the so-called combiner who on behalf of the cooperating group computes the secret. The combiner is successful only if the reconstruction group has at least t members.

Definition 1. *Assume that secrets belong to the set K and shares are from the set S. A (t, n) threshold scheme is a collection of two algorithms. The first algorithm called the dealer*

$$D : K \to S_1 \times S_2 \times \cdots \times S_n$$

assigns shares to the participants for a random secret $k \in K$. Every participant $P_i \in P$ gets his/her share $s_i \in S_i$. If all share sets S_i are equal we simply say that $s_i \in S$. The second algorithm (the combiner)

$$C : S_{i_1} \times S_{i_2} \times \cdots \times S_{i_j} \to K$$

takes shares and computes the secret. The combiner recovers the secret only if the number j of different shares is equal to or bigger than t ($j \geq t$). It fails if the number j of shares is smaller than t ($j < t$).

5.2 General Idea

Assume we have a system containing N mobile agents, denoted as a_i, $0 \leq i \leq N$, L hosts, denoted as h_j, $0 \leq j \leq L$ and a manager, denoted as M.

The manager is similar to the trusted third party: he distributes among agents data needed for proper authentication and also distributes among hosts the data needed for validating agents.

Our system is basically built from two pieces: an authentication method, which can be based on a zero-knowledge proof system or a secret sharing scheme and the modified Merkle's puzzles which provide agents with anonymity.

The proposed system has two phases: the initial one and the authentication phase between an agent and a host.

Initial Phase. At the beginning the manager computes the authentication data: AD. The agents will use this to authenticate themselves to hosts. The manager can create different data for different hosts or different security levels; it depends on the specific security requirements of the system. After creation this data is "wrapped" into a puzzle: the manager creates $G((K, AD), R_i)$, where K is a constant, as described in section 5.1 and R_i is a puzzle. The whole set of puzzles P is now distributed among agents: each of them gets its own subset p_i, such that:

$$P = p_1 \cup p_2 \cup \ldots \cup p_N$$

and for each w, q: $w, q \in \{0, \ldots, N\}$, $w \neq q$

$$p_w \cap p_q \neq \emptyset \text{ and } p_w \neq p_q.$$

This means that the subsets are not disjunctive: a single authentication data can be used by different agents. But the subset assigned to each agent is unique.

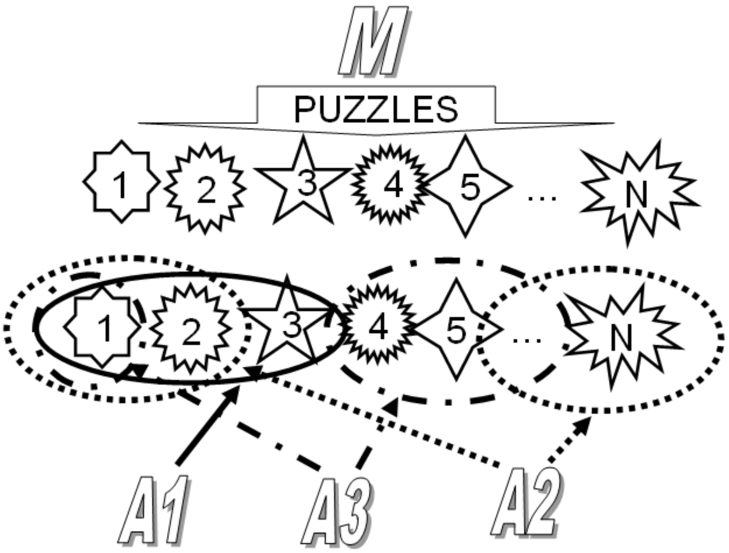

Fig. 1. Initial phase – the puzzles' distribution

Authentication Phase. This takes place when an agent wishes to authenticate itself to a host.

1. An agent sends a randomly selected puzzle to a host.
2. The host solves the puzzle (as described below) and extracts the authentication data.
3. The host checks if the data is proper. He can validate it basing on the information received from the manager. Depending on the underlying authentication method additional steps can be required and the structure of the authentication data and method of validation can differ.
4. If the puzzle (or the extracted authentication data) was previously used within the current host, it asks for another puzzle. If the next verification is successful and the puzzle was not used before, the host provides the agent with required resources or services. If the second puzzle was also already used, agent gets the third chance. If this time the puzzle was also used, the authentication phase fails and has to be started from the beginning.

The Puzzle Generation: Details. As previously stated we utilise Merkle's puzzles (5.1). We propose as G function the DES cipher (or other symmetric cipher) used with a key of approximate length 32 bits. So, the brute-force attack would require approximately 2^{32} computations. The number of puzzles should be of the same order. Using such parameters, a simple brute-force method can be used to solve a single puzzle in a reasonable time and still solving all possible puzzles is computationally hard. Concluding, the puzzles in our system have the following form: $DES((K, AD), R_i)$, where K is a constant and R_i is a key of a symmetric cipher.

5.3 The System Based on the Secret-Sharing Scheme

The first detailed version of our proposal is based on the secret sharing scheme. A brief introduction to secret sharing schemes is in section 1. Our system is utilising the Asmuth and Bloom secure secret sharing scheme ([16]). The secret authentication-message is divided into *n*-parts: *t-1* parts are for host, the rest of them are distributed to agents. The threshold for the secret is *t*. When an agent comes to a host, it is authorised to perform its actions because it has the *t*'th part of secret and he can reconstruct it cooperating with the host.

Initial Phase. The manager randomly chooses n prime or co-prime numbers (called public moduli): p_i ($i = 1, \dots, n$, $p_0 < p_i < \dots < p_n$). They are publicly known. Then, he (playing a role of a dealer in the secret sharing scheme) selects at random an integer s, such that $0 < s < \prod_{i=1}^{t} p_i$. He computes the secret (denoted as k): $k \equiv s \pmod{p_0}$ and shares: $s_i \equiv s \pmod{p_i}$. There have to be at least t participants to recreate the secret. The shares for agents and any additional data are wrapped into puzzles: $DES((K, s_i), R_i)$. The $t-1$ shares are sent to hosts via secure channels. This enables every host to recreate a secret with at least one agent. The manager can create one or more secrets for each host, which later can be used to provide agents with different privileges.

Authentication Phase. The figure below shows general steps of authentication phase in a scheme based on the secret sharing scheme.

(1) Agent $\xrightarrow{puzzle}$ Host
(2) Host extracts s_i from the puzzle
(3) Host recreates secret k using his shares and the agent's share
(4) Host $\xrightarrow{\text{is } k \text{ a valid secret}}$ Manager

1. When an agent wants to authenticate itself to a host, it sends a puzzle of the form:
$$DES((K, s_i), R_i).$$
Also some other additional data can be included within puzzle (e.g., to which secret this share belongs if the host has more than one).
2. The host solves the puzzle and extracts s_i.
3. All t shares, owned by the host and the agent are used to recreate the secret. The host or the manager can act here as a combiner. The secret can be computed by solving the following system of equations:

$$s_{i_1} \equiv s \,(\mathrm{mod}\, p_{i_1})$$

$$\vdots$$

$$s_{i_t} \equiv s \,(\mathrm{mod}\, p_{i_t}).$$

This system has unique solution according to the Chinese Reminder Theorem.

4. After recreating the secret the host has to validate it: he sends it to the manager. Another method of validation is making the secrets public.
5. If the agent's share was previously used, he asks agent for next puzzle as described in the previous section.

The secret can be known to the agent and then be used to provide secure communication between the agent and the host. Alternatively, the host can sent a secret to the agent to authenticate itself.

5.4 The System Based on the Zero-Knowledge Proof

This proposal is not directly based on the zero-knowledge proof, but on the identification system based on zero-knowledge proof. We choose the GQ scheme ([8]) because it is most convenient for our purposes. In this scheme the manager has a pair of RSA-like keys: a public one K_P and a private one k_p. The manager also computes the public modulus $N = p{\cdot}q$, where p, q are RSA-like primes. For the keys, the following equation is true:

$$K_P \times k_p \equiv 1 \,(\mathrm{mod}\, (p-1)\cdot(q-1)).$$

The pair (K_P, N) is made public, k_p, p and q are kept secret. The keys can be used for different purposes, not only for our system.

Initial Phase. First the manager computes set of so-called identities, denoted as ID_p, and their equivalents denoted as J_p. It does not matter how J_p is obtained, provided it is obvious for all participants how to obtain J_p from ID_p. The pairs (ID_p, J_p) are public and can be distributed among hosts. The manager wraps the ID_p into the puzzles $(DES((K, ID_p), R_i)$ and computes also secret value for each ID_p:

$$\sigma_p \equiv J_p^{-k_p} \,(\mathrm{mod}\, N).$$

Each σ_p is distributed with a corresponding puzzle to agents.

Authentication Phase. The diagram below shows general steps of the authentication phase.

$$\begin{aligned}
&(1)\ \text{Agent} \xrightarrow{puzzle,u} \text{Host}\\
&(2)\ \text{Agent} \xleftarrow{b} \text{Host}\\
&(3)\ \text{Agent} \xrightarrow{v} \text{Host}\\
&(4)\ \text{Host validate } v
\end{aligned}$$

1. An agent wanting to authorise itself to a host sends him a puzzle with an identity and a challenge. This challenge is a number computed basing on a random value r, $r \in \{1, \ldots, N-1\}$. It is computed according to:

$$u = r^{K_P} \pmod N.$$

2. After receiving the challenge the host chooses a random value $b \in \{1, \ldots, N\}$ and sends it to the agent.
3. Next, the agent computes the v value basing on the number received from the host and on the agent's secret value σ:

$$v \equiv r \times \sigma^b \pmod N.$$

4. The host uses information extracted from the puzzle, ID_p to obtain J_p and verifies if v is a proper value. To validate the response from the agent, the host checks if

$$J_p^b \times v_{K_P} \equiv u \pmod N.$$

   If the equation is true than the agent proved that he knows the proper secret and should gain an access to the specified resources or services.

As in the previous scheme, the manager can compute many identities, which will give agents different kinds of access to hosts. The second phase can be repeated several times to reduce the probability of cheating the host.

6 Security of the Proposed Schemes

The main purpose of the presented schemes is to provide agents with anonymity and still enable them to securely and efficiently authenticate themselves to hosts. Now we will review our proposal from this perspective. We will assume that one or more hosts are malicious, what means they want to identify agent basing on authentication data (e.g., in order to discover his route or to use this data for other purposes). If there is a group of malicious hosts we assume that they are working together sharing any received information.

In the first system, based on a secret sharing scheme, each agent has several shares of the same secret. So, basing on a value of the share, which was sent by the agent in a puzzle, it is impossible to identify the agent that used it, even if it will be used again with this host or any other. Another method for protecting agents anonymity are puzzles: host can easily extract proper information from one puzzle but even for a group of hosts it is infeasible to solve all existing puzzles and identify the agent basing on his unique subset of authentication data.

Also in the system based on the zero-knowledge proof, a host is incapable to retrieve all existing subsets of identities. Even if the agent is using the same identity again, it will probably select other challenge value, so there is no way for host to differ one agent from any other. What differs this scheme from the previous one is that the agent does not have to show the host its secret value to authenticate itself. It can still be kept secret. This is the main advantage of the zero knowledge proof systems.

The necessity of providing the host with a next puzzle if the current one was already used was introduced to prevent the playback attack by the malicious host against an agent. In our system it is infeasible for any host to compute proper authentication data without agent's cooperation. Without the proposed mechanism the host could use the data sent by an agent to masquerade, playing a role of the agent to some other host or to hold the agent responsible for an action that never took place in reality. Another solution of this problem could be utilising timestamps on puzzles.

The authentication systems used in these schemes are well known and secure. There is no need of discussing their security here. Proofs of this can be found in many publications, e.g. [16].

The manager plays in the system a role similar to TTP, so attacks with cooperation of the manager are not discussed in this paper.

To provide agents with full security an additional integrity mechanism should be used. Some are described in [10] and [20].

The proposed schemes enable agents to preserve anonymity and securely authorise themselves to different hosts. They also provide confidentiality of agents route.

7 Conclusions

Recent developments in mobile technology open new fields for applications of mobile agent systems. In future, agents will need to have the same possibilities in the Internet as we have in the real world to act on our behalf. However, preserving anonymity and providing security is still a main issue in many agent systems. In this paper we proposed new authorisation systems, enabling agents to stay anonymous. The presented systems are based on a certain secure secret-sharing scheme and, alternatively, on some zero-knowledge proof. These systems are an effective way of providing the security and the anonymity for mobile agents. The proposed solution is easy to implement in many existing agents' systems, making possible a secure and anonymous communication between agents and other parties.

References

1. Ashley, P., Au, R., Looi, M., Seet, L.T.: Secure Authorisation Agent for Cross-Domain Access Control in a Mobile Computing Environment. ICICS 2001, LNCS 2288, pp. 369–381, Springer 2002
2. Beimel, A., Dolev, S.: Buses for Anonymous Message Delivery, Journal of Cryptology, Volume 16, Number 1, January 2003, 25–39
3. Berkovits, S., Guttman, J.D., Swarup, V.: Authentication for Mobile Agents, in: Mobile Agents and Security, LNCS 1419, pp. 114–136, Springer 1998
4. Cachin, C., Camenisch, J., Kilian, J., Muller, J.: One-Round Secure Computation and Secure Autonomous Agents, Automata, Languages and Programming, pp. 512–523, 2000
5. Camenisch, J., Lysyanskaya, A.: An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation, EUROCRYPT 2001, LNCS 2045, pp. 93–118, Springer 2001
6. Chan, P.K., Fan, D.W., Lee, W., Prodromidis, A.L., Stolfo, S.J., Tselepis, S.: Jam: Java agents for meta-learning over distributed databases. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, 1997

7. Corradi, A., Cremonini, M., Montanari, R., Stefanelli, C.: Mobile Agents Integrity for Electronic Commerce Applicatons, *Information Systems* Vol. 24, No. 6, pp. 519–533, 1999
8. Guillou, L., Quisquater, J.-J.: A Practical Zero-knowledge Protocol Fitted to Security Microprocessor Minimizing both Transmission and Memory. Proceedings of Eurocrypt 88, Springer-Verlag Eds, pp. 123–128, 1988.
9. Jansen, W.: Determining Privileges of Mobile Agents, NIST (www.nist.gov)
10. Jansen, W., Karygiannis, T.: NIST Special Publication 800–19 – Mobile Agents Security
11. Chrissikopoulos, V., Katsirelos, G., Kotzanikolaou, P.: Mobile Agents for Secure Electronic Transactions, in N.E. Mastorakis, editor, Recent Advances in Signal Processing and Communications, pp. 363–368. World Scientific Engineering Society, 1999
12. Kulesza, K., Kotulski, Z., Kulesza, K.: On Mobile Agents Anonymity; Formulating Traffic Analysis Problems, in: Advanced Computer Systems, Proceedings of the 10th International Conference, ACS'2003, Miedzyzdroje, October 22th–24th 2003, pp. 15–21.
13. Kulesza, K., Kotulski, Z.: Decision Systems in Distributed Environments: Mobile Agents and Their Role in Modern E-Commerce, in: A.Lapinska, [ed.] Information in 21st Century Society, University of Warmia and Mazury Edition, Olsztyn 2003, pp. 271–282. ISBN 83-89112-60-4.
14. Margasinski, I., Szczypiorski, K.: VAST: Versatile Anonymous System for Web Users. Tenth International Multi-Conference on Advanced Computer Systems ACS'2003, Miedzyzdroje, Poland, October 2003, published as a book chapter by Springer-Verlag, 2004
15. Merkle, R.: Secure Communications over Insecure Channels, in: Communications of the ACM, April 1978 (pp. 294–299).
16. Pieprzyk, J., Hardjono, T., Seberry, J.: Fundamentals of Computer Security, Springer, Berlin 2003
17. Reiter, M. K., Rubin, A. D.: Crowds: Anonymity for Web Transactions, ACM Transactions on Information and System Security, Vol. 1, No. 1, November 1998, pp. 66–92.
18. Reyes, A., Sanchez, E., Barba A.: Routing Management Application Based on Mobile Agents on the INTERNET2. EUNICE 2000, Holland
19. Wangham, M.S., da Silva Fraga, J., Obelheiro, R.R.: A Security Scheme for Mobile Agent Platforms in Large-Scale Systems. CMS 2003, LNCS 2828, pp. 104–116, Springer 2003
20. Zwierko, A., Kotulski, Z.: A new protocol for group authentication providing partial anonymity. NGI 2005 – In Proc of: The First EuroNGI Conference – Traffic Engineering for the Next Generation Internet, 18–20 April 2005, Rome, Italy (accepted)