An Untraceability Protocol for Mobile Agents and Its Enhanced Security Study

Rafał Leszczyna[1] and Janusz Górski[2]

[1] European Commission, Joint Research Centre,
Via Enrico Fermi, 21020 Ispra (VA), Italy
[2] Gdansk University of Technology,
Narutowicza 11/12, Gdańsk, Poland

April 12, 2006

Abstract

Among various computational models to accommodate the mobile wireless computing paradigm, the mobile agent model has major technological advantages. Moreover, our research shows that the model has some unique features which, if used appropriately, may support user privacy. We therefore propose a security protocol for mobile agents aiming at user untraceability. We have performed a basic security analysis of the protocol and evaluated its performance. We have also implemented it, made it available to users, and applied it to an e-health service. In this paper, we present an enhanced security study of the protocol. As far as we know, such systematised study has not yet been performed. Additionally, known analyses limit their interest to particular attackers (internal attackers). We believe that the attack checklist which we developed to evaluate our protocol, may serve as a helpful reference for authors of other protocols. The paper is also addressed to designers and administrators of mobile agent systems who consider provision of user untraceability within their systems.

1 Introduction

To accommodate the new computing paradigm introduced by mobile wireless computing, various software models have been proposed including the client/agent/server, the client/intercept, the peer-to-peer, and the Mobile Agent (MA) models (Spyrou, Samaras, Pitoura, & Evripidou, 2004).

Among them the MA model appears to be particularly suitable due to its technological advantages including bandwidth conservation, latency reduction, disconnected operation, load balancing and dynamic deployment (Gray, Kotz, Cybenko, & Rus, 2000; Chess et al., 1995). These features correspond to characteristics of mobile wireless networks, which, compared with traditional fixed

networks, present lower transfers, less reliable and more expensive communications, lower computational power, more lightweight devices and flexible locations of participating devices (Spyrou et al., 2004).

Furthermore, (Spyrou et al., 2004) presents a MA framework able to instantiate dynamically each of the software models. The authors claim that their MA implementation of the models further enhances their applicability to mobile wireless computing, making applications more light-weight, tolerant to intermittent connectivity and adaptable to dynamically changing environments.

The primary focus of our research into MA, has been methods for their protection, since security is one of the biggest challenges in the field (Jansen & Karygiannis, 2000). In the progress of this research, we discovered that Mobile Agent Systems (MAS) have some characteristics which may facilitate protection of users' privacy (Leszczyna & Górski, 2004). Based on the observation we proposed a security protocol aiming at anonymity (precisely – untraceability) of agent users (Leszczyna & Górski, 2005b, 2004). We evaluated its performance (Leszczyna & Górski, 2005a) and performed a basic security study (Leszczyna & Górski, 2004). We implemented the protocol (Leszczyna, 2005b) as an open source package (the package is available at `http://jade.tilab.com/community-3rdpartysw.htm#Untraceability`) and deployed it in an e-Health anonymous counselling scenario (Leszczyna, 2005a).

In this paper we present an extended security study of the protocol.

To prepare this study we applied known Traffic Analysis (TA) attacks to MAS and discussed their feasibility. This way a list of attacks against untraceability protocols for MA was created, against which we then validated our protocol.

The study shows that certain TA attacks are very potent against untraceability protocols. For them we present additional measures which can be introduced to increase the level of security.

As far as we know, such systematised study has not been performed until now since previous analyses limit their interest to particular attackers (internal attackers). We note that the attack checklist may also serve as a helpful reference for authors of another protocols. The part focusing on additional measures in case of effective attacks is addressed to designers and administrators of MAS who might consider employing an untraceability protocol.

In the next subsections we present a notion of anonymity and introduce a minimal set of fundamental requirements for software agents required to understand the paper. Next we briefly describe the protocol and finally move on to the security analysis.

1.1 Anonymity

Anonymity is the property that ensures that a user may use a resource or a service (the *items of interest*) without disclosing their identity (National Institute of Standards and Technology (NIST), 1998). *Untraceability* is a subclass of communication anonymity (Pfitzmann & Hansen, 2005) assuring that the identity can not be inferred by tracing a message (an agent – in case of MAS).

Anonymity plays a crucial role in many activities conducted in the Internet. For example users may be reluctant to engage in web-browsing, message-sending and file-sharing, unless they receive guarantees that their anonymity will be protected to some reasonable degree (Halpern & O'Neill, 2004). (Gülcü & Tsudik, 1996) describe four categories of internet applications where anonymity is required[1]: discussion of sensitive and personal issues, information searches, freedom of speech in intolerant environments and polling/surveying.

1.2 Mobile (Autonomous) Agents

Since no established taxonomy of agents exists, we support the notion of generic *software agents* being software programs acting on behalf of users (this is an etymologically inspired definition, see (Harper, 2006; Merriam-Webster Incorporated, 2006)). Then we use adjectives to describe a particular type of agent which is acting in our interest (after the (Franklin & Graesser, 1996) and (Murch & Johnson, 1999) classifications). In this article we are in principle concentrated on mobile autonomous agents i.e. those able to exercise control over their own actions (*autonomous*) and able to roam networks freely (*mobile*).

Generally an agent's structure is described in terms of a *state* and a *behaviour*. Roughly speaking, the term behaviour refers to an agent's code and the state to the data held by the agent.

Mobile agents are given a *goal* and to accomplish it they roam from one network node (in MAS, called a *container*) to another, starting from a *base*. The sequence of containers to be passed is called a *route*. The agents' goal can be formulated through explicit indication of a node to be visited or in a more abstract way, without indicating nodes, for instance: to provide information about the cheapest vendor of a certain product. In the former case the node is called a *destination*. Note that in the case of 'abstract' goal formulation, the identity of receiver is naturally obscured.

1.3 The Protocol

In common scenarios, agents cover a route and come back to their base. To allow them to return, the identifier of the base must be explicitly stored in their state. Our protocol aims at obscuring this identifier.

Very few proposals have been made thus far for MA untraceability (Goldschlag, Reed, & Syverson, 1996; Westhoff, Markus Schneider, Unger, & Kaderali, 2000; Enzmann, Kunz, & Schneider, 2002) and these are based on conception of onion routing (Reed, Syverson, & Goldschlag, 1998), which requires a route determination before launching an agent. Unfortunately such an approach restricts spontaneous route selection during migration by an agent, an advantage over traditional message-based communication, which is required for certain applications (targeting dynamic deployment, load balancing etc (Gray et al., 2000)). Our protocol doesn't introduce this limitation.

[1] It is important to note that the authors don't claim this set to be exhaustive.

In brief, the concept of the protocol is that at each visited container an agent asks the container to encrypt the identifier of previous container using the current container's secret key. Then the encrypted identifier is put into the LIFO queue stored in the agent's state. After achieving its goal, when the agent wishes to come back to its base container, it uses the queue to find its way back. Down the route back, the identifiers are subsequently decrypted by containers using their secret keys. For a more formal specification of the protocol, refer to our earlier work (Leszczyna & Górski, 2005b).

W ten sposb agent ma wolno w wyborze trasy podczas gdy idzie w kierunku celu. Natomiast powrt ma ograniczony, co jest cen za untraceability. Studium performance pokazuje to, wskazujc jednoczenie, e jest to gwny element complexity i nie zoono wynikajca z uycia kryptografii.

2 Security Study

Cryptographic Protocol Verification (CPV) methods spam into two groups: formal and informal.

The informal methods such as evaluation based on a list of known attacks (which we discuss later in this section) or attack-oriented testing (testers try to break the security of protocols), are clear and straightforward, but they don't give full warranties of security. What they say is that a protocol is safe from *certain* attacks. However a new attack can always appear, to which the protocol will be prone.

For this reason some people consider using formal methods more appropriate. Formal methods for CPV (a nice overview of which can be found in works of Meadows, e.g. (Meadows, 2003a, 2003b)) are based on a model of the protocol and the context of its use (usually: its users (principals); a communication protocol (communication channels and messages exchanged), an attacker and security measures) and its verification. Formal methods aim at providing full mathematical proofs of correctness and to achieve this goal mathematical constructs are employed such as logics and algebras. Unfortunately, in practice formal models of protocol contexts are often too complex to allow their verification (and often also their creation) in a reasonable timescale, thus some simplifying assumptions are usually taken. In consequence warranties given by formal approaches also tend to be limited.

Apart from a few fields where particular approaches tend to fit extraordinarily well (e.g. formal methods seem to work well with authentication protocols) the most common opinion is that there is no silver bullet and different complementary methods should be used.

In the case of untraceability protocols the context is so complex that they are particularly difficult to approach formally (Meadows, 2003a). Attacker modelling seems to be an especially demanding task. The adversary can compromise communication media (*external*) and/or mix nodes, recipients and senders (*internal*) (Raymond, 2001). The adversary may succeed in attacking all nodes (*omnipresent*) or `k` of them (*k-listening* (Dolev & Ostrobsky, 2000)) *actively*

or *passively*. An active adversary can arbitrarily modify the computations and messages (by adding and deleting) whereas a passive adversary can only listen (Raymond, 2001). Finally different configurations, hybrids and alliances of attackers may occur, such as external-active or colluding internal and external.

Such an extensive model of an attacker means that so far most security analysis of untraceability protocols is based on a list of known attacks, which is performed through taking each known attack and referring it to the proposed protocol. Additionally some evaluators deliver probabilistic assessments which aim at providing a wider picture of a protocol's security (Kesdogan, Egner, & Büschkes, 1998; Goldberg & Wagner, 1998; Reiter & Rubin, 1998).

Some formal frameworks were proposed for untraceability protocols (e.g. (Syverson & Stubblebine, 1999; Garcia, Hasuo, Pieters, & Rossum, 2005)), however they are not supported by automatic checkers. This imposes a need for manual validation and makes verification a demanding task (for a notation must be learned). Also an ordinary user may feel quite bewildered when presented such a mathematical proof method.

As far as verification of untraceability protocols for mobile agents is concerned, until now evaluation based on a list of attacks has been performed, but this has been limited to models of internal attackers, which we consider as insufficient. We also conducted such a study, which helped to expose an attack and motivated us to present measures to harden the protocol against resistant it[2], but this had to be followed by further studies taking into account other types of attackers.

In this section we present the results of such extended evaluation. We intend to accompany this with more formal analyses in the future.

The analysis shows that an attacker aiming at compromising the anonymity of agent owners has available such a large variety of attack paths that it is impossible to defend it using only an untraceability protocol.

Other means (e.g. dummy traffic, agent delaying) must be introduced into the system in which communication anonymity is to be preserved (which will result in an *untraceability architecture*). In this paper, when discussing particular attacks, we recall these countermeasures. We decided not to include the measures in the protocol specification, giving freedom to designers and administrators to decide whether to increase the level of anonymity at the cost of efficiency.

Because both attacks and countermeasures are also relevant to other untraceability protocols for software agents(Goldschlag et al., 1996; Westhoff et al., 2000; Enzmann et al., 2002), the study, which originally was dedicated only to our protocol, in the end gives more general conclusions applicable to other protocols.

To construct the list of attacks against untraceability protocols for MAS we reviewed anonymity bibliography available at `www.freehaven.net/anonbib/`. Of the prominent studies we reviewed (e.g. (Kesdogan et al., 1998; Goldberg

[2]Not without a price. Because the attack is specific, requiring a particularly strong attacker, so it needs correspondingly, strong (and costly) countermeasures. Thus we decided to propose an alternative protocol (Leszczyna & Górski, 2004; Leszczyna, 2005a).

& Wagner, 1998; Reiter & Rubin, 1998; Mazières & Kaashoek, 1998; Dolev & Ostrobsky, 2000; Raymond, 2001)), Raymond's appears to be the most comprehensive (it actually summarises all the others). Thus we took Raymond's work and discussed all the attacks it describes step by step referring them to MAS.

Note that different attacks assume different types of adversary, so we don't state a general attacker model. It should rather be considered on a case by case basis depending on the attack.

2.1 Attacks Based on the Agent's Distinguishing Features

If an external attacker is able to read an agent state (which usually is the case), they can easily recognise and trace the agent through subsequent containers, as long as the state is easy distinguishable from the states of other agents. Thus, in such cases, obfuscation of the agent state is required. This is performed in such a way that the state is different before and after traversing each container. Because to provide such means of state obfuscation is normally out of the scope of untraceability protocols for MA, it is up to designers and administrators to consider various cryptographic schemas for this obfuscation e.g. encryption using a public key of the next station.

2.2 Brute Force Attack

If the obfuscation is introduced, a powerful (either omnipresent or able to move to all places in the network) external attacker can follow every possible route the agent could have passed through. They observe the agent entering a container and then follow all outgoing agents. They can do so for each of the agents on and on until the destination is reached.

If it can be guaranteed that containers are always visited by a certain number of agents and more than one leaves a container at the same moment, then the probability of linking a base and a destination decreases exponentially with the length of the route. However it may happen that a network has low traffic and at a particular instant, the agent was the only one which traversed a given route. Then it is totally exposed to an adversary.

To avoid this, in traditional networks, dummy traffic is usually introduced. This is a costly option, overloading the network, narrowing its bandwidth etc. But in the case of MAS, it is more worthwhile. In MAS all nodes are equal in the sense that there is no distinction between mixes and ordinary nodes, yet all nodes are mixes. So the attacker cannot easily detect which node is the destination one. They might guess that it is the last one before returning to the base, but it is enough to force the agent to traverse additional containers on its way back to prevent such inferences.

To hide the base station, one option is to introduce 'redundant' agents which start their lifecycle on random containers and then roam around the network waiting for a task. Thus when a user needs an agent they pick it up rather than launch. Because the agent started its lifecycle earlier on a container different to

the one where it is being picked up, this previous container is the base container and not the one where the agent was employed.

This means of protection is not as strong as the introduction of dummy traffic, because the probability decreases linearly with the route length, but it is much less costly and may be sufficient, especially if the network is naturally loaded with normal agent traffic.

2.3 Timing Attacks

If traversing a route of a particular length always takes an agent a specific time, then an external adversary may correlate times taken by observed agents. Raymond, for example (Raymond, 2001) assumes two routes to cover, for which subsequently 1 second and 3 seconds are always required. Then if two messages are observed arriving in the network at 0:00 and 0:01 and leaving it at 0:03 and 0:02, discovering which message passed which route is trivial.

In MAS, timing attacks are much less successful due to the difficulty in identifying the destination (see section 2.2). To increase this already high level of anonymity it is worth considering the introduction of random delays of agents visits to containers or batch processing (see the next section).

Note, that in practical agent environments, in which agents roam and perform tasks, random delays may naturally result from the tasks.

2.4 The Node Flushing Attack (a.k.a. Spam Attack, Flooding Attack, n-1 Attack)

In traditional networks a popular method of protection against TA is batch processing – a mix waits until `n` messages are collected before flushing them. This method protects from timing attacks. This is done at the price of message delays, which in case of reasonable network traffic may be acceptable if we take into consideration the quality received in return.

One attacker response is the *n-1 attack* in which they 'fill' the mix with `n-1` of their messages, so the mix will be left with `n-1` messages which are well distinguishable to the adversary and separated from message of interest. Thus we must assure that recognising their messages (agents) is not an easy task for an adversary. For this, we recommend to use the obfuscation method mentioned in section 2.1 together with nonces.

2.5 Contextual Attacks

These are attacks related to communication habits of users. Raymond distinguishes between three general types of attacks in this group: Communication Pattern Attacks, Packet Counting Attacks and Intersection Attack (Raymond, 2001).

All these attacks are applicable to MAS and, as with the traditional networks – they are difficult to protect from. This is mainly because they rely on actors being out of the control of system designers and administrators.

Communication Pattern Attacks – refers directly to the users' habits such as time of using network services and thus the user must not to have habits which are too unusual, if they want to remain anonymous.

Packet Counting Attacks – in this case, a user makes the task of tracing them easier by launching an agent of a characteristic, distinctive size (in terms of number of packets). To avoid this, a standard size of agent may be used, but this leads either to costly redundancies if the size is large, or to inconvenient boundaries, if the size is too small, imposing the distribution of data between more than one agent. Generally, protection from packet counting attacks seems to be hardly achievable using systemic methods.

Intersection Attacks – where an adversary observes network traffic and stepwise narrows the range of possible interlocutors (as described in (Berthold, Pfitzmann, & Standtke, 2000)). Such attacks actually undermine the defence of using *different* routes each time an agent goes to the same destination. Imagine an agent travelling twice from a base `A` to a destination `B`, each time passing through completely different containers. If an adversary observes these two trips, they notice that the only containers in common are `A` and the `B`, which makes them good candidates for interlocutors.

In conclusion, counteracting contextual attacks must be the user's responsibility: users should be conscious of the fact that by performing characteristic activities, they become more distinct and thus easier to trace.

2.6 Denial of Service Attacks

An attacker compromises some mix-nodes (making them inoperative), counting on fact that it will force a user to send their messages through them to change their behaviour. Though this attack works in the case of conventional networks, when applied to MAS, it appears to be ineffective, since agents arbitrarily choose a different route each time they roam. In case some containers are nonfunctional, they simply omit them and choose other. We just have to make sure that an agent will not behave abnormally when the destination container is compromised. It means that the agent should perform the whole route as if nothing had happened (not quickly return to its origin to report the damage).

2.7 Active Attacks Exploiting User Reactions

An example of such an attack is message interception and broadcasting to all possible recipients. The idea is that the original recipient will behave differently from the others. To prevent this kind of attacks, as in case of DoS attacks (see section 2.6), the feature of completing the whole route even in unusual situations (this time it is that the declared destination appears not to be the real one) should be included.

2.8 Agent Delaying

In this case, an external attacker stops an agent until he gathers enough network resources or until the network becomes easier to monitor or to see if possible recipients receive other messages, etc. To protect from this attack, administrators should consider introducing authenticated timing information aiming at detection of unwanted delays.

2.9 Agent Tagging

An adversary marks an agent, to facilitate its tracing.

Feasibility of attacks where distinct data are used for marking is very dependant on a particular system's architecture and should be discussed in relation to a real environment. In most cases, because of network protocol characteristics, such tagging is not feasible on the network layer, what makes the attack unavailable to an external attacker.

An internal attacker may perform it, however because of the above (see section 2.2) mentioned similarity between mixes and ordinary stations in MAS, it becomes ineffective.

Other known tagging attacks are *Agent Delaying* and *Shadowing*.

Agent Delaying – in contrast to (see section 2.8), this attack aims at distinguishing an agent in the network. This can be achieved through ensuring that the agent stops at each container for a specific, characteristic (but not suspect) time. This attack is difficult to prevent, but fortunately, also to perform, since it requires an external omnipresent attacker with the ability to modify the agent behaviour.

Shadowing – an attacker intercepts messages and copies them. After this, `k` copies of the message traverse the same route. Effective in traditional networks, in MAS (and using our 'autonomy-friendly' protocol) it appears to be useless since each of the copies autonomously chooses its own route. Thus it is important to make sure that agents choose their routes in a nondeterministic way. Deterministic routing forms a serious vulnerability – if adversaries can discover its algorithm they could predict agent routes.

2.10 Corrupted Party Attacks

Taking over the sender does not lead to any gain in knowledge because of the autonomous routing feature sustained by our protocol.

The interesting case is when the destination party is corrupted. According to (Raymond, 2001), the attack assumes that an adversary is able to encode some indicative information into a message (an agent – in our case) thus making it distinct to others. Raymond accompanies his description with two examples: varying the reply latency and sending messages of a specific length. With this description the attack appears to be a tagging attack performed by an active internal adversary, which we have already discussed in section 2.9.

3 Conclusions

We performed the enhanced security study, which in our opinion extends the analyses provided so far for untraceability protocols of MA.

To perform the study we discussed known TA attacks for traditional networks and, based on this discussion, prepared a list of TA attacks against MA. The list may be a helpful reference in security studies of other protocols.

The analysis shows that some features of MAS support anonymity-friendliness. Primarily we mean the difficulty in identifying the base station, which stems from the fact that all containers are potential mixes. Also autonomous routing, which, in contrast with other known untraceability protocols for MA, is preserved in ours, helps avoid many attacks.

On the other hand, we indicate points where a powerful enough (e.g. external) attacker may be successful in their attempt to trace an agent. Here, we recall that the decision of whether to introduce countermeasures or not should be left to system designers and administrators.

References

Berthold, O., Pfitzmann, A., & Standtke, R. (2000, July). The disadvantages of free MIX routes and how to overcome them. In H. Federrath (Ed.), *Proceedings of designing privacy enhancing technologies: Workshop on design issues in anonymity and unobservability* (pp. 30–45). Springer-Verlag, LNCS 2009. (`http://www.tik.ee.ethz.ch/~weiler/lehre/netsec/Unterlagen/ano%n/disadvantages_berthold.pdf`)

Chess, D., Grosof, B., Harrison, C., Levine, D., Parris, C., & Tsudik, G. (1995). Itinerant Agents for Mobile Computing. *IEEE Personal Communications*, *2*(5), 34–49. (`citeseer.ist.psu.edu/article/chess95itinerant.html`)

Dolev, S., & Ostrobsky, R. (2000). Xor-trees for efficient anonymous multicast and reception. *ACM Trans. Inf. Syst. Secur.*, *3*(2), 63–84. (`www.cs.ucla.edu/~rafail/PUBLIC/31.ps`)

Enzmann, M., Kunz, T., & Schneider, M. (2002, July). Using mobile agents for privacy amplification in the trade with tangible goods. In *6th world multi-conference on systemics, cybernetics and informatics (sci)* (Vol. IV). Orlando, Florida, USA.

Franklin, S., & Graesser, A. (1996). Is it an agent, or just a program?: A taxonomy for autonomous agents. In *Intelligent agents III. agent theories, architectures and languages (ATAL'96)* (Vol. 1193). Berlin, Germany: Springer-Verlag.

Garcia, F. D., Hasuo, I., Pieters, W., & Rossum, P. van. (2005). Provable anonymity. In *Fmse '05: Proceedings of the 2005 acm workshop on formal methods in security engineering* (pp. 63–72). New York, NY, USA: ACM Press. (`http://doi.acm.org/10.1145/1103576.1103585`)

Goldberg, I., & Wagner, D. (1998, August). TAZ servers and the rewebber network: Enabling anonymous publishing on the world wide web. *First Monday, 3*(4). (`http://www.ovmj.org/GNUnet/papers/goldberg97taz.ps`)

Goldschlag, D. M., Reed, M. G., & Syverson, P. F. (1996, May). Hiding Routing Information. In R. Anderson (Ed.), *Proceedings of information hiding: First international workshop* (pp. 137–150). Springer-Verlag, LNCS 1174.

Gray, R. S., Kotz, D., Cybenko, G., & Rus, D. (2000). *Mobile agents: Motivations and state-of-the-art systems* (Tech. Rep. No. TR2000-365). Hanover, NH: Dartmouth College.

Gülcü, C., & Tsudik, G. (1996). Mixing email with Babel. In *Proceedings of the 1996 symposium on network and distributed system security (sndss '96)* (p. 2). IEEE Computer Society.

Halpern, J. Y., & O'Neill, K. R. (2004). Anonymity and information hiding in multiagent systems. *Journal of Computer Security.* (`http://www.cs.cornell.edu/people/oneill/papers/jcs_halpern_o%neill.pdf`)

Harper, D. (2006). *Online etymology dictionary.* Website. (`http://www.etymonline.com/` (last access: May 4, 2005))

Jansen, W., & Karygiannis, T. (2000). *Nist special publication 800-19 - mobile agent security.* (`citeseer.ist.psu.edu/jansen00nist.html`)

Kesdogan, D., Egner, J., & Büschkes, R. (1998). Stop-and-go MIXes: Providing probabilistic anonymity in an open system. In *Proceedings of information hiding workshop (ih 1998).* Springer-Verlag, LNCS 1525.

Leszczyna, R. (2005a, September). The solution for anonymous access of it services and its application to e-health counselling. In *Proceedings of the 1st 2005 ieee international conference on technologies for homeland security and safety (tehoss '05)* (Vol. 1, pp. 161–170). Gdańsk, Poland: Gdańsk University of Technology.

Leszczyna, R. (2005b, September). *Untraceability i add-on for jade.* Via Enrico Fermi 1, Ispra, Italy. (The add-on is available to download at `http://jade.tilab.com/community-3rdpartysw.htm#Untraceability` (last access: December 16, 2005))

Leszczyna, R., & Górski, J. (2004, December). *Untraceability of mobile agents* (Tech. Rep.). European Commission, Joint Research Centre, Institute for the Protection and security of the Citizen.

Leszczyna, R., & Górski, J. (2005a, October). *Performance analysis of untraceability protocols for mobile agents using an adaptable framework.* (Accepted for 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '06), Future University-Hakodate, 8 – 12 May 2006)

Leszczyna, R., & Górski, J. (2005b, July). Untraceability of mobile agents. In *Proceedings of the 4th international joint conference on autonomous agents and multiagent systems (aamas '05)* (Vol. 3, p. 1233-1234). Utrecht, the Netherlands.

Mazières, D., & Kaashoek, M. F. (1998, November). The Design, Implementation and Operation of an Email Pseudonym Server. In *Proceedings of the 5th ACM Conference on Computer and Communications Security*

(CCS 1998). ACM Press. (ftp://cag.lcs.mit.edu/pub/dm/papers/mazieres:pnym.pdf)

Meadows, C. A. (2003a, January). Formal methods for cryptographic protocol analysis: emerging issues and trends. *Selected Areas in Communications, IEEE Journal on*, *21*(1), 44-54. (Formal methods for cryptographic protocol analysis: emerging issues and trends; Meadows, C.; Naval Res. Lab., Washington, DC; This paper appears in: Selected Areas in Communications, IEEE Journal on; Publication Date: Jan 2003; On page(s): 44- 54; Volume: 21, Issue: 1; ISSN: 0733-8716)

Meadows, C. A. (2003b). *Towards a hierarchy of cryptographic protocol models.*

Merriam-Webster Incorporated. (2006). *Merriam-Webster Online.* Website. (http://www.m-w.com/ (last access: May 4, 2005))

Murch, R., & Johnson, T. (1999). *Intelligent software agents.* Prentice Hall PTR.

National Institute of Standards and Technology (NIST). (1998). *Common criteria for information technology security evaluation - part 2: Security funtional requirements.* U.S. Government Printing Office.

Pfitzmann, A., & Hansen, M. (2005, December). *Anonymity, unlinkability, unobservability, pseudonymity, and identity management a consolidated proposal for terminology (version v0.25).* Website. (http://dud.inf.tu-dresden.de/Anon_Terminology.shtml (last access: February 15, 2006))

Raymond, J.-F. (2001). Traffic analysis: Protocols, attacks, design issues and open problems. In H. Federrath (Ed.), *Designing privacy enhancing technologies: Proceedings of international workshop on design issues in anonymity and unobservability* (Vol. 2009, pp. 10–29). New York, NY, USA: Springer-Verlag New York, Inc. (citeseer.ist.psu.edu/454354.html)

Reed, M. G., Syverson, P. F., & Goldschlag, D. M. (1998). Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, *16*(4).

Reiter, M., & Rubin, A. (1998, June). Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, *1*(1). (http://avirubin.com/crowds.pdf)

Spyrou, C., Samaras, G., Pitoura, E., & Evripidou, P. (2004, October). Mobile agents for wireless computing: The convergence of wireless computational models with mobile-agent technologies. *Mob. Netw. Appl.*, *9*(5), 517–528. (http://doi.acm.org/10.1145/1027347.1027354)

Syverson, P. F., & Stubblebine, S. G. (1999, September). Group principals and the formalization of anonymity. In *Proceedings of the world congress on formal methods* (Vol. 1, pp. 314–333). Springer-Verlag, LNCS 1708. (citeseer.ist.psu.edu/syverson99group.html)

Westhoff, D., Markus Schneider, I., Unger, C., & Kaderali, F. (2000). Protecting a mobile agent's route against collusions. In *Proceedings of the 6th annual international workshop on selected areas in cryptography* (pp. 215–225). Springer-Verlag.