VU VRIJE UNIVERSITEIT AMSTERDAM

VU Research Portal

Organized anonymity in agent systems

Warnier, M.E.; de Groot, D.R.A.; Brazier, F.M.

published in
Informal Proceedings of the Fourth European Workshop on Multi-Agent Systems (EUMAS'06)
2006

document version
Publisher's PDF, also known as Version of record

Link to publication in VU Research Portal

citation for published version (APA)
Warnier, M. E., de Groot, D. R. A., & Brazier, F. M. (2006). Organized anonymity in agent systems. In *Informal Proceedings of the Fourth European Workshop on Multi-Agent Systems (EUMAS'06)*

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

E-mail address:
vuresearchportal.ub@vu.nl

Download date: 26. Apr. 2025

ORGANIZED ANONYMITY IN AGENT SYSTEMS

Martijn Warnier David de Groot Frances Brazier

Intelligent Interactive Distributed Systems
Faculty of Sciences, Vrije Universiteit Amsterdam
{*warnier,davidra,frances*}*@cs.vu.nl, http://www.iids.org*

Abstract

Anonymity is of great importance in distributed agent applications such as e-commerce & auctions. This paper proposes and analyzes a new approach for organized anonymity of agents based on the use of pseudonyms. A novel naming scheme is presented that can be used by agent platforms to provide anonymity for each individual agent. The paper introduces two distinct techniques, one based on handles and another based on agent spawning. Both techniques can be integrated into agent platform middleware, automatically guaranteeing anonymity for all individual agents. The applicability of this approach is evaluated for three agent platforms: AgentScape, JADE and SeMoA.

1 Introduction

Agent techniques provide state-of-the-art solutions for distributed applications such as e-commerce, e-health, resource negotiations and electronic auctions [11]. Anonymity of agents can be an important requirement for such applications. This can be acquired by conventional methods such as the use of a mediator or another outside trusted third party. A mediator or trusted third party acts on behalf of an agent (and its owner) and relays messages without an agents identity. However, such methods require an explicit effort on the agent application developer. He/she not only needs to interpret a mediator in his/her application, (s)he also needs to make sure that no information regarding the agent's identity is leaked by other means. Only if a number of necessary precautions are taken by the developer can agents not be tracked and are they thus anonymous.

This paper proposes a new approach for anonymity for agents and agent based applications that guarantees anonymity for all agents without any additional effort on the part of agent application developers. Assuming agents trust the middleware –the agent platform– on which they run, anonymous communication between agents can be guaranteed.

The focus of this paper is on the anonymity of individual agents. The link between agent and owner does not have to be anonymous, since the middleware is trusted and can thus be trusted to keep this information confidential[1]. Anonymity is not an absolute notion, one can be anonymous to one person and not to another. Similarly, agents can be anonymous to other agents, but not to the agent platform, as assumed in this paper. Several degrees of anonymity can be distinguished ranging from absolute anonymity to total non-repudiation [3, 6]. The paper focuses on *organized semi-anonymity*. It is both organized and semi-anonymous: the agent platforms issues pseudonyms, which provide anonymity for an agent with respect to other agents, but not with respect to the agent platform itself.

A naming scheme is introduced that ensures that an agent's true identity and its pseudonyms are unique and cannot be linked to each other by any outside parties. Two techniques that provide this form of communication anonymity are presented: one based on pseudonyms and one based on agent spawning. Both techniques can be integrated into the agent platform. This hides them for agent (developers) while ensuring anonymity automatically.

[1] We think that, from a practical perspective, it is reasonable to assume that most agent platforms are unwilling to accept agents that are completely anonymous. Such agents can run havoc on agent systems and the agent platform is unable to take any (legal) action against the agent owner, nor can it recognize future malicious agents coming from the same source. This is clearly an unwanted situation, therefore identification of agent owners by agent platforms, as well as other identity management mechanisms for agent systems [4], form a prerequisite in this paper.

Our approach provides a dedicated and organized solution for anonymity of agents, in contrast to more general anonymizing techniques. Most notably, Korba, Song and Yee [8] use onion routing [5] for all inter-agent communication. Onion routing provides anonymous communication by redirecting messages via a number of routers using an unpredictable path. Their approach is implemented in the JADE agent platform [2] where a dedicated communication layer facilitates all anonymous inter-agent communication. This approach can guarantee the same level of anonymity as our own. Our approach however forms a dedicated solution for agents that can be fine-tuned to meet a range of agent specific settings.

The distinct advantages and (minor) disadvantages of our approach to organized anonymity are analyzed, and illustrated in three agent platforms: AgentScape [12], JADE and SeMoA [15].

An overview of anonymity as commonly defined in Computer Science and the application of this definition to distributed agent systems in Section 2 provides the background for this paper. Section 3 gives a high-level overview of the naming scheme for anonymity. Section 4 presents two different techniques that can be used to acquire anonymity. Section 5 discusses a number of ways how these techniques can be realized, ranging from complete integration in the agent platform middleware to solutions based on individual agent implementations. Section 6 discusses the actual implementation of the approach for anonymity in three agent platforms. The papers ends with discussion and conclusions.

2 Anonymity in agent systems

Classical anonymity in computer systems focuses on anonymity of the underlying communication layer [16], as does this paper. The typical goal is to anonymously browse the Internet, or communicate with other parties without revealing the parties true identity. The related notions of anonymity are:

- sender anonymity
- receiver anonymity
- link anonymity (unlinkability)

The first two, sender and receiver anonymity, require that the location of the sender and receiver, respectively, are hidden for the other communicating party. Link anonymity, also known as unlinkability, ensures that the link between the communicating parties remains anonymous to all third parties: it is impossible for any outside party to observe if two parties are communicating with each other. Note that the communicating parties themselves are often aware of the (true) identity of the other party. In practice these forms of anonymity can be combined.

This general notion of anonymity in computer science can be applied to agent technology. The focus is on anonymity of individual agents. It is assumed that the relation between agent owner and agent is confidential, and guaranteed by the agent platform. Full anonymity, i.e., receiver, sender and link anonymity taken together, can only be established when an agent cannot be linked to a legal entity, either directly or indirectly via its communication with other agents. A platform based solution that enables the middleware to (automatically) provide link anonymity and location anonymity for each individual agent is the main focus of this paper. Pseudonyms are used for this purpose.

The use of a pseudonym on its own does not suffice. Outsiders can possibly observe communication events of an agent and use this to obtain (unwanted) information about an agent. Another danger occurs if an agent uses the same pseudonym to communicate with several other agents. Together these agents can infer that they have been talking to the same party, breaking anonymity.

In agent systems that support mobility of agents yet another form of anonymity exists: migration anonymity. In essence this hides the migration path of an agent, and thus hides the original starting platform of an agent which results in a form of anonymity. Migration anonymity falls outside the scope of this paper, Rafal Leszczyna and Janusz Górski [10] discuss migration anonymity in more detail.

3 A naming scheme for anonymity

An agent platforms can identify an agent by its *globally unique identifier*(GUID). Such a GUID corresponds uniquely with the identity of the agent. Pseudonyms can be used for communication, as stated above. Using only one pseudonym is not enough to obtain link anonymity. Consider the following example:

Example 1
There are three agents A, B and C, each with their own pseudonym P_A, P_B and P_C respectively. Agent A considers obtaining services from either agent B or C. It first uses its pseudonym P_A to communicate with agent B and asks agent B the price of its service, then agent A uses the same pseudonym P_A to ask agent C the price of its services. Although agent B and agent C do not know A's real identity, together they can still determine that the same agent *has been asking price information of the services they provide. Thus agent A has not been communicating anonymously.*

Example 1 above clearly shows the need for agents to use a pseudonym for each individual communication event (or communication session) to obtain (link) anonymity. For similar reasons, agents should also use a different pseudonym each time they communication with the same party at a later time interval. This ensures that an agent does not reveal too much information over time which can compromise link anonymity.

In our approach each agent has one globally unique identifier and has multiple pseudonyms that are used for each separate communication event.

An agent platform should provide a naming scheme that ensures that all GUID's and pseudonyms are unique. Moreover, GUID's must be hidden from other agents and pseudonyms cannot be linked to a specific agent and to each other.

4 Techniques for anonymity

There are several techniques that can be used to realize the naming scheme for anonymity. This section discusses two such techniques.

The first technique shows how the naming scheme can be acquired using handles. A handle is a unique and meaningless string [1] that is bound to a specific agent. This works well, but it sometimes requires some modifications to the underlying agent platform middleware. The other technique, based on agent spawning, can be readily implemented in several current agent platforms without any modifications to the platforms' middleware. The disadvantage of this is that agent developers have to choose to use this technique since it is not provided automatically by the platform. Section 6 shows how the techniques from this section can be implemented in a number of agent platforms.

4.1 Using handles for anonymity

Each agent is assumed to have a global unique identifier (GUID) which is only known to the agent platform. Such a GUID can, for example, be implemented by a Universally Unique Identifier (UUID, ISO 11578:1996). Furthermore, each agent can acquire as many (globally unique) handles as it requires. These handles serve as pseudonyms and are used for communication purposes.

Since handles have no intrinsic meaning and they do not leak any information about an agent or its owner agents can safely use handles as pseudonyms. Link anonymity can be acquired by agents if they use a new handle for each communication event.

The agent platform is responsible for handing out agent GUID's and handles. Handles and GUID's should be related. An agent platform should be able to check if the handle that an agent uses indeed belongs to it. However, others (agents in particular) should not be able to determine if two handles belong to the same agent. This can be accomplished by using a cryptographic hash function [7]. The following algorithm is used by the agent platform to generate handles for agents, where 'sha' is a cryptographic hash function:

$\text{handle}_1 = \text{sha}(\text{GUID} + 1)$
$\text{handle}_2 = \text{sha}(\text{GUID} + 2)$
...
or more general:

$\text{handle}_n = \text{sha}(\text{GUID} + \text{n})$ with $n \in \mathbb{N}^+$

This has two specific advantages:

- If the GUID is not known then handles cannot be linked to each other or one specific GUID.
- If the GUID is known then the platform cannot deny that a specific handle belongs to a GUID.

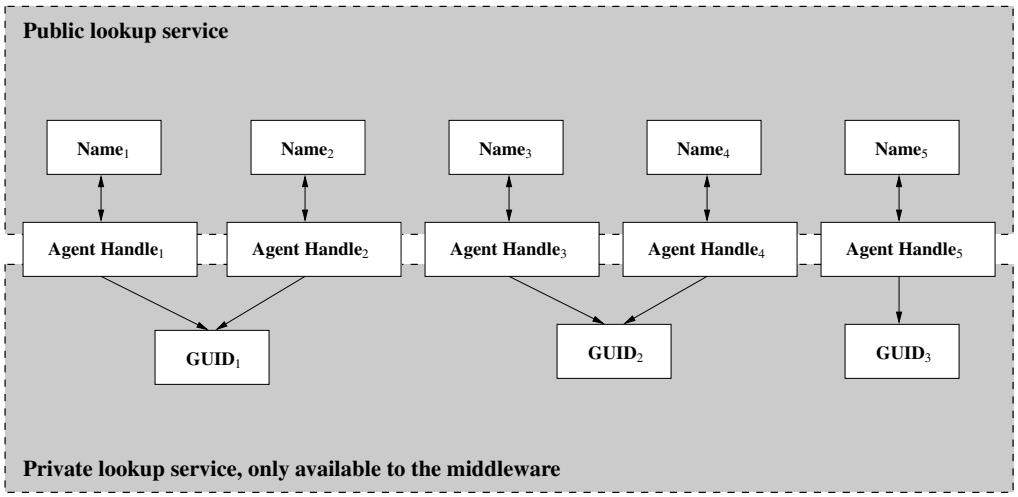

Figure 1: Schematic overview of the handle technique

Several properties of cryptographic hashes, such as Sha-1 and MD5 are used: First the fact that cryptographic hashes are *one way* is used, i.e., easy to compute but very hard (computably infeasible) to reverse. This guarantees that if an agent knows a handle it can not compute the corresponding GUID. Cryptographic hashes also have the property that they are *collision free*, i.e., it is computably infeasible to construct two different GUID's that have the same hash-image. This ensures that all handles are unique. And cryptographic hashes also have the property that a *small change* in the input results in a big change in the output (roughly half of the bits should change). This ensures that agents can not determine if two handles belong to the same GUID (agent).

A lookup service is necessary so that platform locations can link handles back to GUID's. It is essential that the lookup service that maps handles to GUID's is *private* to the middleware. If this is the case then link anonymity for agents can be guaranteed. Agents can however also use an optional human readable name, where each name corresponds with one handle. A *public* lookup service also links names with handles and vice versa. Figure 4.1 schematically displays the handle technique, where arrows indicate a possible lookup.

Since handles are used for all communication between agents, where handles are only unique meaningless strings, no information about the location of a particular agent is revealed. Hence this technique also provides location anonymity and thus also sender and receiver anonymity. Whenever two agents communicate they do not have to share the location (host) on which they reside.

Agents can implicitly revoke a handle (simply by no longer using it) or explicitly (in which case the handle is removed from the lookup service). By definition, handles are also unique over time, due to the large number of possible handles no handle is ever used twice. However, if so required, a time-stamping mechanism can be used to limit the lifetime of individual handles and hence make handles applicable for reuse. Note that guaranteeing uniqueness of handles is important for reliability, correctness and accountability.

4.2 Using agent spawning for anonymity

Another technique can be used to acquire anonymity. For this technique to work the agent platform should allow agents to spawn their own (other) agents and it should not be possible to determine which agents are related via spawning. Thus if an agent is spawned then other agents should not be able to tell whether the agent really is a spawned agent and which agent created it.

If these requirements are met then agents can use spawned agents for communication purposes. In essence, a spawned agent is used as a pseudonym for its original agent. The spawned agent works as a simple proxy whose sole task it is to redirect messages to other agents and back to the originating agent. In principle, an agent can spawn a new agent for each communication event and hence can be (link) anonymous with respect to other agents. Note that in this technique spawned agents are basically used as handles. Again, the agent platform is the only party that may be aware of the link between spawned and 'original' agents, which is similar to the private lookup service in the technique explained in Section 4.1.

Location anonymity can also be provided using this technique, provided that agents are either allowed to

create agents on other platforms or that agents can migrate between platforms. As a dedicated agent is used for communication purposes only other agents can determine on which host the spawned communicating-agent resides (if the platform discloses this information). On which platform the original (anonymous) agent that started the communication event resides is not known to other agents.

Whether location anonymity can be provided also depends on the platform, e.g. in JADE-S [14] (a secured extension of JADE) one can require that a pre-defined format for agent names is used. This format is defined by a local administrator by means of a security policy. For example, the security policy, as proposed in the JADE-S documentation, can specify for a user named Bob that he can only create agents if the name contains the prefix '`bob-`', resulting in a name such as '`bob-agent1@platformA.myexample.org`'. If a security policy of this form is used then agent spawning cannot be used for anonymity.

5 Middleware implementation

The techniques described above can be implemented at several levels, ranging from middleware to services to individual agents.

An organized solution for anonymity requires a solution that is *integrated in the middleware* of the agent system. This allows agents to communicate anonymously without any additional effort by the agent developer. The handle technique can easily be integrated completely in the middleware. A new handle is automatically generated by the agent platform for each communication event. Since agents are not part of the middleware, the agent spawning based approach can not be combined with the middleware.

However, it is often not necessary to make the communication with all parties anonymous. In fact sometimes an agents needs to reveal its identity to gain access to certain information or an agent might want to establish several (virtual) identities. Such cases require a more fine-grained solution.

Policies can be defined on a per agent basis when an agent's communication should be anonymous. More advanced policies, e.g. self-learning or self-organizing ones [18], can refine this further and allow agents to be anonymous to some agents while not anonymous to others.

An *anonymizing service* –implemented by an agent– on top of the middleware forms another implementation possibility. A dedicated agent works as a sort of proxy and routes all communication. As long as this proxy agent uses a new handle or spawned agent for each new communication event the communication is anonymous. This 'gateway' approach has the obvious advantage that agent developers have a more fine tuned control of anonymity. The anonymizing service can be used if circumstances require this.

Self management by the agent developer is the most finely tuned approach. Agent developers can implement anonymity themselves using either handles or spawning techniques. This allows control on a very fine scale, but it also requires the most effort on the agent developers part.

5.1 Observations

Both the handle and the agent spawning technique have advantages and disadvantages.

The 'handle' technique of Section 4.1 has the advantage that it provides an organized solution for all agents that can easily be integrated into the agent platform middleware. Location anonymity is also guaranteed as all communication is routed via handles that do not reveal any information about an agent's location. Finally, the performance overhead of this technique is marginal. The largest disadvantage is that the technique cannot be implemented in most existing agent platforms without significant changes to the platform's middleware.

The 'spawning' technique of Section 4.2 has as its largest advantage that it can be used without any changes to the agent platform's middleware, provided that the platform supports agent spawning and that creating and spawned agents cannot be linked. A disadvantage is that the solution is not organized, i.e., can not be completely integrated into the agent platform middleware. A dedicated anonymizing service –implemented by a spawning agent– forms a reasonable alternative. An other disadvantage is that the performance overhead is significant: instead of a handle the middleware has to create a new agent for each communication event which obviously consumes some additional platform resources. Location anonymity, can only be provided if agents are allowed to either spawn agents on other locations or are allowed to migrate between locations.

When to use which technique depends on circumstances. In cases where only a limited number of agents require anonymity, the spawning technique probably make more sense. The performance penalty outweighs

the convenience of using a familiar agent platform. However, if all agents require anonymity then the handle technique is better, even if this means a partial re-implementation of an existing agent platform.

6 Evaluation of existing agent platforms

The techniques discussed in the previous section are analyzed for three agent systems: AgentScape [12], our own agent platform, JADE [2], the most broadly used agent system and SeMoA [15], an agent system that focuses on security.

The handle-based approach is (partially) implemented in AgentScape as is the agent spawning approach. The other platforms (JADE and SeMoA), require significant changes to the agent platform middleware to implement handles. This section only discusses the agent spawning techniques, that can be implemented on top of the middleware, for these platforms

6.1 AgentScape

AgentScape[2] is a framework for development and deployment of open, large-scale distributed agent systems and includes support for fault-tolerance, security, heterogeneity and interoperability [12]. AgentScape's middleware security features include separate use of globally unique identifiers (GUID's) and handles (of agents and services), leasing of resources, sandboxing of agents, signing agent's code and its state, and secure communication. In AgentScape both techniques (use of handles and agent spawning) for implementation of anonymity can be applied.

Handles

AgentScape uses a handle-model as described in Section 4.1. Each agent has its own globally unique identifier (GUID). The GUID is generated upon creation of the agent and is kept private to the middleware. An agent always has an initial handle that can be linked —by the middleware– to the agent's GUID. Optionally, a name can be assigned to a agent's handle. Additional handles can be requested to the middleware. An agent's handles and names are registered in a Name Look-up Service (NLS), that is assumed to be private to the middleware[3]. An anonymizing service and automatic generating of new handles for each communication event are not (yet) integrated in the AgentScape middleware. However, implementation of these modules should be straightforward and can be expected in a future release of AgentScape.

Spawning

AgentScape supports agent spawning. Furthermore, spawning and creating agents cannot be linked to each other, thus the spawning based technique from Section 4.2 can be used for anonymity.

6.2 JADE

JADE[4] (Java Agent Development Environment) is a development-framework and runtime execution environment for distributed multi-agent systems [2]. JADE is FIPA-compliant, i.e., it adheres to the FIPA standard[5], which are intended to promote the interoperability of heterogeneous agents and services.

The agent spawning technique can be used for anonymity in JADE. Agents can easily start new agents and choose names for them. A possible threat for an agent's anonymity is formed by the Agent Management Service (AMS). Every agent is required to register in the Agent Management Service. This registered information is visible to others. Carefully monitoring the AMS can potentially break the anonymous communication of agents, especially in situations where there are only a small number of agents on a platform.

[2] Version 0.9.0beta1, available at `http://www.agentscape.org`

[3] Two different lookup services can be used in AgentScape: the *default lookup services*, that is completely open and accessible to anybody and *a secure, decentralized lookup service* [17] that is only accessible to the AgentScape middleware.

[4] Version 3.4, available at `http://jade.tilab.com`

[5] `http://www.fipa.org`

6.3 SeMoA

SeMoA[6] (for "Secure Mobile Agents") is an agent platforms that has as main goals to provide a secure agent environment and to develop an extensible and open server for mobile agents [15] and to provide a secure platform for mobile access to multimedia data and services based on mobile agent technology. SeMoA supports agent spawning, thus self-managed and dedicated-agent that provide anonymity can be implemented.

In SeMoA, policies determine permissions for creation and deletion of agents (and many other operations). Since policies are specific for a single agent or agents of a specific agent owner (and valid for a certain server), using a dedicated agent for communication is possible, because the communication-dedicated agent can be started by another (trusted) agent owner. It is also possible to run a dedicated anonymizing agent in another SeMoA server where a different policy does allow agent spawning.

7 Discussion

This paper introduces a new anonymizing approach for agent systems that guarantees organized semi-anonymity for each individual agent. Two distinct techniques can be used to realize anonymity for agents. The technique based on handles is completely integrated into the agent platform middleware. The technique based on agent spawning cannot be integrated into the middleware, but this technique can easily be deployed in current agent platforms, as AgentScape, JADE and SeMoA, without any changes to the underlying middleware.

Under which conditions which technique should be used depends on circumstances. In cases where only a limited number of agents require anonymity, the spawning technique discussed in Section 4.2 makes most sense. The performance penalty outweighs the convenience of being able to use a broad range of agent platforms. However, if all agents require anonymity then the handle technique described in Section 4.1 is more promising, even if this requires a partial re-implementation of an existing agent platform.

Although the approach discussed is theoretically secure, in practice a number of risks remain. If the number of agents on an entire agent platform is known then, in theory, it is possible that all agents conspire together against one agent. This breaks link anonymity. A simple solution to this problem is to use a number of 'dummy' agents that belong to the agent platform itself. Other agents cannot determine if an agent is real or belongs to the platform an thus the attack no longer works.

Another risk, although highly unlikely due to its intrinsic properties, is the use of side channel attacks [9]. Timing attacks, as discussed in [13], form a particular challenging problem. Using a combination of techniques that observe timing behaviour together with statistical analysis almost certainly breaks anonymity.

If an organized solution based on handles is preferred the agent platform of choice is AgentScape. The three analyzed platforms –AgentScape, JADE and SeMoA– all support agent spawning based solutions for anonymity. More insight in the performance overhead of the handle techniques in AgentScape is needed. Current research focuses on this aspect.

Acknowledgments

This research is conducted as part of the ACCESS project[7] funded by the NWO TOKEN program. The authors also thank Stichting NLnet for their support.

References

[1] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. A layered naming architecture for the internet. *SIGCOMM Comput. Commun. Rev.*, 34(4):343–352, 2004.

[2] F. Bellifemine, A. Poggi, and G. Rimassa. JADE–A FIPA-compliant agent framework. *Proceedings of PAAM*, 99:97–108, 1999.

[3] F. Brazier, A. Oskamp, J. Prins, M. Schellekens, and N. Wijngaards. Anonymity and software agents: An interdiscplinary challenge. *AI & Law*, 1-2(12):137–157, 2004.

[6] Version June 26, 2006, available at `http://www.semoa.org`

[7] `http://www.iids.org/access`

[4] D. de Groot and F. Brazier. Identity Management in Agent Systems. In N. Foukia, J. Seigneur, and M. Purvis, editors, *Proceedings of the First International Workshop on Privacy and Security in Agent-based Collaborative Environments (PSACE)*, pages 23–34, 2006.

[5] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium*, volume 2, 2004.

[6] J. Grijpink and J. Prins. New rules for anonymous electronic transactions? An exploration of the private law implications of digital anonymity. In *Digital Anonymity and the Law, Tensions and Dimensions, Information technology and law series*, volume 2, pages 249–269. T.M.C.Asser Press, 2003.

[7] C. Kaufman, R. Perlman, and M. Speciner. *Network Security, PRIVATE Communication in a PUBLIC World*. Prentice Hall, 2nd edition, 2002.

[8] L. Korba, R. Song, and G. Yee. Anonymous Communications for Mobile Agents. In *Proceeding of the 4 thInternational Workshop on Mobile Agents for Telecommunication Applications (MATA'02)a*, volume 2521 of *LNCS*, pages 171–181, 2002.

[9] B. W. Lampson. A note on the Confinement Problem. *Communications of the ACM*, 16(10):613–615, 1973.

[10] R. Leszczyna and J. Górski. Untraceability of mobile agents. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 1233–1234, New York, NY, USA, 2005. ACM Press.

[11] M. Luck, P. McBurney, and C. Preist. *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)*. AgentLink, 2003.

[12] B. Overeinder and F. Brazier. Scalable middleware environment for agent-based internet applications. In *Proceedings of the Workshop on State-of-the-Art in Scientific Computing (PARA'04)*, volume 3732 of *Lecture Notes in Computer Science*, pages 675–679, Copenhagen, Denmark, June 2004. Springer.

[13] A. Pashalidis and C. Mitchell. Limits to Anonymity when Using Credentials. In *Proceedings of the 12th International Workshop on Security Protocols*, LNCS. Springer-Verlag, 2004.

[14] A. Poggi, M. Tomaiuolo, and G. Vitaglione. Security and trust in agent-oriented middleware. In R. Meersman and Z. Tari, editors, *OTM Workshops*, volume 2889 of *Lecture Notes in Computer Science*, pages 989–1003. Springer, 2003.

[15] V. Roth and M. Jalali-Sohi. Concepts and architecture of a security-centric mobile agent server. In *"Proc. of the Fifth International Symposium on Autonomous Decentralized Systems (ISADS 2001)"*, pages 435–442. IEEE Computer Society, 2001.

[16] A. Serjantov. *On the anonymity of anonymity systems*. PhD thesis, University of Cambridge, 2004.

[17] R. van Schouwen. Design and implementation of a secure, decentralized location service for agent platforms. Master's thesis, Department of Computer Sciences, Vrije Universiteit Amsterdam, aug 2006.

[18] A. Weimerskirch and G. Thonet. A Distributed Light-Weight Authentication Model for Ad-hoc Networks. In *The 4th International Conference on Information Security and Cryptology (ICISC 2001)*, volume 2288 of *LNCS*, pages 341–354. Springer, 2002.