Proceedings of the *Fifth IEEE Symposium on Computers and Communications: ISCC 2000*, Antibes, France, 3-6 July 2000, p. 160–165

Security Requirements for Management Systems using Mobile Agents

H. Reiser
Munich Network Management Team
University of Munich, Germany
reiser@informatik.uni-muenchen.de

G. Vogt
Munich Network Management Team
Munich University of Technology, Germany
vogt@in.tum.de

Abstract

Flexible and distributed management systems based on mobile agents have certain advantages over centralized and static management architectures. However, security plays a decisive role in terms of acceptance and applicability of mobile agents. In this paper we analyze the threats and attacks against mobile agent systems used for management purposes. Therefore, general models of mobile agent based management systems are developed. Based on a risk analysis of these models we derive security requirements. In order to satisfy these requirements components and services are identified and integrated in a comprehensive security architecture.

Keywords:
Security, Mobile Agent Systems, Management by Delegation

1. Introduction

Mobile agents[1] are a new paradigm in distributed systems that allow transfering not only simple data but also 'living' code through networks. Therefore, an agent system provides a homogeneous run–time environment for agents by adapting the underlying, heterogeneous host system. In addition, it offers general services to agents making them easier to handle and smaller. Although the main research on mobile agents does not focus on its applicability in management, several publications [2, 5, 7, 18, 20] regard it as a promising approach. First, mobile agents give a more generic view on some aspects of concepts like Management–by–Delegation (MbD) [8, 22]: In terms of mobility, delegation is migration of a mobile agent from a management server to a mid–level–manager or managed resource. Second, mobility may overcome limitations of MbD as it allows mobile agents to migrate: A mobile agent is not limited to remain on a managed resource after delegation. In fact, it can decide autonomously to move to another place, for example for load balancing or to apply some complex operations on a group of resources. Third, management systems can use mobile agents for implementing distributed management functionality.

Although mobile agents have many benefits for distributed computing they introduce a new dimension of security issues. Automatically executing arbitrary code on any host can be dangerous. The same care is necessary as if manually starting programs from unknown sources. In order to protect hosts from malicious code, agent systems usually provide a virtual machine or interpreter to run mobile agents in a separate, locked environment. Any action or communication of agents is then only possible through the means of the agent system (similar to applets in WWW browsers).

But this covers only a single aspect of security. In order for mobile agents to fulfil their management tasks they must be able to access security sensitive data and resources. This must happen in a controlled manner and only by mobile agents that are allowed to do so. A closer look on security reveals various threats in different areas. Many of them have been identified [3, 10, 27]. For some of them possible solutions have been presented. For some of them there are ideas how they might be solved (e.g. authentication [1], access control [4], trust [6], secure MbD [23], securing mobile agents from malicious hosts [13, 17, 21, 26]).

As most solutions and ideas only deal with a single problem they remain fragments. However, making mobile agent technology secure means to integrate these fragments in an architecture. Moreover, in order to get a complete view of possible threats there is a need for an overall model that allows identifying and examining all points of attacks.

In this paper we look into security issues of agent systems under the special constraints of management systems. Although a 'general–purpose' agent system might be used, it is still questionable if it will meet the needs of a management system. Whereas access to a general agent system is usually open to the public, e.g. the systems run agents from unknown sources, we consider this a bad idea for management. Dealing with vital devices and systems, tight security must prevent any misuse. Therefore, agent systems depend on certain trust in other agent systems and agents, i.e. there is always a person liable for an action. We find this a major distinction to 'general–purpose' agent systems.

In the next section we propose two models that describe the security-related aspects of mobile agent based management systems. They allow us to find points of attack and to deduce possible threats. The analysis of threats and the classification of attacks follows in section 3. Instead of developing a defense strategy for each possible attack, the generalization into security requirements in section 4 is a better approach. Section 5 presents a security architecture for mobile agent based management systems. The last section concludes the paper presenting issues for further research.

2. Architecture Models

Relations are the main idea relevant for security in this paper. A relation between two entities exists if they exchange any kind of information. An intruder either attacks an existing relation or an existing entity. In the latter case, he must

[1] Throughout this paper we use the term 'agent' for the autonomous, mobile unit, the subject of this paper. It is not to be confused with management agents as an interface of a managed resource to management systems.

establish a relation with his target. If there are no relations with others, an entity is isolated and thus not subject to attacks (certainly even isolated systems may be subject to attacks. However, this requires physical access to the system and is thus not considered in this paper). If there are relations security may become important.

To recognize attacks it is necessary to identify relations and entities participating in these relations. Knowing these entities it is possible to classify relations as specific forms of interactions. Finally, the functional components of these entities must be identified. These components are the actual points of attacks and thus have to implement security.

2.1. Entities — Organization Model

Entities are the main participants in mobile agent based management systems. In order to find all relations a model of the management system is necessary that helps to understand possible roles of entities with their relations. Therefore, we start with any management architecture (e.g. central, hierarchical or multicenter control management [11]) and introduce agent systems with mobile agents in existing relations of traditional management entities. The results reveal that there are several different kinds of entity pairs and relations. Figure 1 is a simplified but general example of all entities

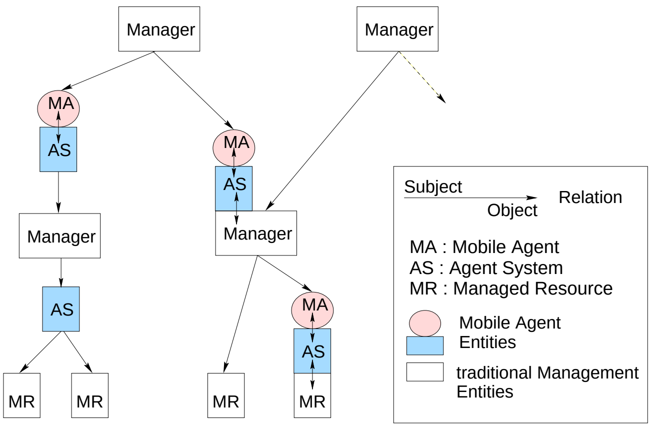

Figure 1. Possible Organization Model

and relations. It shows entities as subjects (on higher levels) acting on objects (on lower levels). Although it resembles a hierarchical management architecture it is not limited to such architectures. Managers in this figure are not only management stations but also management tools etc. used by management stations, acting on managed resources. A managed resource is any device, system or application to be managed.

Some relations between entities are local to a host system. Adjacent entities represent this case in the figure. Free arrows represent remote communication between entities. Agent systems in the figure may be local to a manager or managed resource but may as well be separated on a different host. This is necessary as we do not expect all managers (e.g. tools) and managed resources to provide full agent systems (i.e. including all security mechanisms). This allows integrating legacy systems while still being able to move mobile agents 'closer', e.g., to a managed resource.

Following roles of entities can be identified:

- Managers as possible sources of mobile agents and as objects of management operations and mobile agents.
- Agent systems as execution platform for mobile agents, possibly running on systems hosting a manager or managed resource.
- Mobile agents as mobile units initiated by a manager, executed by agent systems.
- Managed resources as objects of mobile agent and management operations.

2.2. Interaction and Components Model

Relations represent information or data exchange between two participants. Therefore, a relation can be regarded as an information channel. Moreover, three kinds of relations exist in this model: there are traditional communication relations, i.e. two entities have a relation because they exchange messages of any kind. In addition, using agent systems there are two relevant local relations: execution relations, e.g. a manager executes an agent system that executes mobile agents, and calling relations, e.g. a mobile agent calls interface functions of a mobile agent system or vice versa. As mobile agents are part of the management system it is necessary to examine these relations. Each entity as well as each possible relation between two entities is vulnerable to attacks.

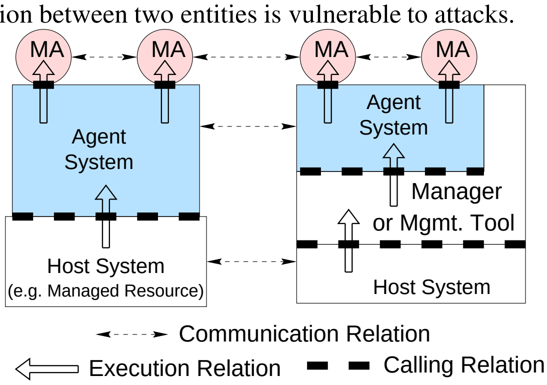

Figure 2. Interaction and components model

These three kinds of relations are the only ways of interaction between entities. The interaction model reflects these kinds of relations from the organization model. Thus, the interaction model consists of three major parts: communication, execution and calling. Figure 2 shows all kinds of relations focusing on the major components in implementations involving agent systems. In this figure dashed, horizontal arrows are communication relations. Following the idea of the OSI basic reference model [15] higher layers use services of lower layers to communicate. Vertical arrows show execution relations. One layer executes the layer above, e.g. the host system runs the agent system that, again, runs mobile agents. Dashed lines between two layers indicate calling relations. As the figure suggests, access to layers below the one beneath should be indirect, e.g. the agent system should guard access of agents to the host system.

Communication relations can be considered as transport of messages, e.g. through a network. On the one hand, both parties run in different environments without direct interference, i.e. no side has direct access to memory or other local resources at the other end. Any action may only happen upon receipt of a message but still under full control of the receiver. On the other hand, the transport media itself or at least the data transmitted requires protection. A network may be complex with several components between both ends, spanning a longer distance.

Adding agent systems to management systems, there are also relations that are different to communcation and need closer attention. With an agent system executing an agent there is an **execution relation** between both. As both run on a single machine even direct manipulation of either one might be possible. The agent system provides the environment as part of the execution relation. Therefore, an agent system may easily spy out data carried by the agent, manipulate the execution trace or even deny execution.

Once the agent comes to life it has a **calling relation** with the agent system as it usually needs various services of the agent system to fulfil its task, e.g. access to base system libraries or network services. However, all interaction must happen in a controlled manner to protect the agent system and the environment. An agent might try to gain unauthorized access to the agent system or host system

Looking closer at entities we find a basic structure as indicated in figure 2 (in a simplified way). A **host system** is any kind of computer or device that takes part in the management system. Main requirements are the abilities to provide network communication and to run further program code. Besides, a host system can have a fully-fledged operating system, but it may be as well a simple network device.

The **agent system** adapts the host system to meet the needs of mobile agents. In respect to the management system it can be considered as infrastructure. It usually uses a virtual machine to provide a common, homogeneous execution platform. Therefore, it hides heterogeneous details of the host system, tries to protect it and fills shortages. Moreover, the agent system offers additional, important services to agents, e.g. naming, location, communication services. Depending on the type of host system an agent system may use underlying mechanisms, e.g. encryption or authorization services, key management or access control.

Mobile agents are actually part of the management application. They use the agent system for their execution and for services they need. As they cannot live without an agent system they also highly depend on protection to be provided by the agent system. Actually, agents are at the mercy of the agent system as it seems almost impossible to protect agents against malicious agent systems [12]. Therefore, the management application must take this into consideration, e.g. it must not send an agent with sensitive data or task to an agent system not fully trusted.

3. Analysis of threats

Various kinds of attacks and threats could compromise the security of mobile agent based management systems. An attack is an attempt to illegally access a system, a resource or information or to execute malicious code. Attacks are classified as active and passive attacks [25]. The ability of an attacker to change something is characteristic for an active attack. In a passive attack he only collects information but does not to actively manipulate an object. In the following we take a closer look at attacks focusing on those to mobile agent based management systems. As mentioned in the last section every entity as well as every relation between two entities can be a target of an attack. Therefore, we distinguish **attacking an entity** from **attacking a relation**.

There are three kinds of relations in the interaction model (figure 2): communication, execution and calling relations. It must be distinguished between attacks which are generally possible for all kinds of relations and those which are special to a particular kind of attack (see figure 3).

<table>
<tr><th>Entity Attacks</th><th colspan="3">Relation Attacks</th></tr>
<tr><td rowspan="3">Masquerade</td><td colspan="2">Eavesdropping
Theft of rights
Repudiation
Replication</td><td>Alteration
Denial-of-service
Resource misuse
Man-in-the-middle</td></tr>
<tr><th>Calling Relation Attack</th><th>Communication Relation Attacks</th><th>Execution Relation Attacks</th></tr>
<tr><td>Circumvention</td><td>Replay
Redirection</td><td>Execution trace manipulation
Denial-of-execution</td></tr>
</table>

Figure 3. Classification of threats and attacks

3.1. Entity Attacks

The organization and interaction models show four main entities which can act as subjects as well as objects: managers, mobile agents, agent systems and managed resources. Each of these entities can be attacked.

The attacker tries to become a 'valid identity' by faking an identity or entity in the management system. This attack is called masquerade, e.g., if the attacker can act as a manager or if he can launch an MA or AS under the name of a legitimate subject he is able to gain illegal access to the system.

3.2. Relation Attacks

In addition to entity attacks, there are also attacks to relations between two legitimate entities. Relations between more than two entities can be split into several two-entity-relations. Some of them apply to relations in general and some of them are specific to a particular kind of relation.

Eavesdropping of messages can enable the attacker to gain information paving the way for further attacks or to steal confidential data. This is a passive attack and very hard to detect.

A management system based on MAs, implementing the MbD–paradigm, must delegate management functions or management tasks to MAs. MAs must also be able to delegate functionality and rights to other subjects. Additionally, a group of MAs must handle a management task in cooperation. For this purpose it is essential to delegate rights or permissions to other subjects. MbD and the delegation of rights make a new kind of attack possible: the theft of rights or delegation misuse. Rights can be stolen during execution or during transmission of an MA.

Any subject must be liable for its sensible and critical actions. It is necessary to identify the user which is responsible for the message or action. For example, it must be impossible to launch an MA doing malicious actions and, afterwards, repudiate everything. This relation attack is called repudiation. Another attack in this regard is the unauthorized replication of MAs. A malicious MA, AS or manager may replicate MAs. Besides, an intruder in a relation may duplicate an MA or message during transmission.

If the attacker can actively manipulate the information channel he can do alterations to messages. In this case, he may change the functionality or data of a migrating agent. The AS is a mediator between MAs and hosting systems. In addition, it provides a runtime environment for MAs. Therefore, a malicious AS can read, alter or delete data of an local MA (alteration of code and data).

An attacker can do a denial–of–service attack against communication relations, an AS or a hosting system, e.g. a hostile MA overloads the attacked resource and thus it is impossible for other legitimate subjects to use the resource. This scenario is even more complicated if the denial–of–service attack is not done by a single MA but by a distributed group of malicious MAs.

Another attack is resource misuse. As an MA implements management functionality and must therefore have administrator rights. The MA can abuse communication resources, resources of the underlying host system or of the AS.

Despite of these general attacks there is one which only affects the **calling relation**. The attacker can try to circumvent the dedicated calling interfaces to directly access other methods not intended to be used. Also the **communication relation** is security sensitive. An attacker may store a message or an MA and send it once more at a later time to a destination. This is called replay attack. Moreover, an attacker can also redirect agents and messages or delay them.

The last kind of relation attack is that against **execution relations**. As an MA can only live with the aid of an agent system it is even feasible for a hostile agent system to manipulate the execution trace of an MA. For example, the AS can manipulate the runtime stack of the MA, prevent execution of a certain function or force execution of additional functionality. Another possible attack is to prevent execution of MAs (denial–of–execution). As an AS has to execute the MA and thus has complete control of the agent, these attacks are almost impossible to prevent. For this reason, we either assume a relationship of trust between delegator/MA and AS or we demand to take this into consideration before migration. On the other hand, a malicious MA can attack the AS, the underlying hosting system or other MAs in various ways (e.g. denial–of–service, resource misuse).

4. Security Requirements

Regarding the various attacks it is possible to develop a defense strategy for each kind of attack. But this approach has the drawback that any new attack requires a new defense strategy and the security system always 'lags behind' the attacker. The more promising approach is to develop a security architecture which implements a more abstract view on attacks. The OSI security architecture [14, 16] may be regarded as a basis, but it must be adapted to particular characteristics of agent systems. The first step towards such an architecture is to deduce a conceptual view on counteractions against classes of attacks: security requirements. In order to satisfy these requirements several components and services have to be identified and integrated in a security architecture for a mobile agent based management system. Such an architecture is able to prevent complete classes of attacks and even future attacks belonging to one of these classes.

The organizational model together with the threat analysis gives a view onto entities. It is essential for a secure management system to be able to identify the subjects and objects representing the participating entities. The security requirement is authentication. Mobile agents are a new kind of access to systems that need closer attention. Some available access control devices like fingerprint scanners may improve control of access to humans but they will not work for mobile agents. Authentication is very fundamental, because most of the following security requirements presuppose the ability to identify subjects and objects unambiguously.

Authorization is necessary to bind rights to subjects. For that purpose rights and permissions must be described. Access control must then enforce rights and restrictions at runtime. Each object in the system offers interfaces which can be used by subjects. Access control prevents illegal access of objects. Certain management tasks require that a mobile agents is able to delegate rights and permissions to other entities, a concept for delegation of these rights is necessary. Security management with the aid of mobile agents can be carried out if such a concept is available.

Each information channel representing a relation between entities may need protection. The security requirement confidentiality is satisfied if such a channel is only accessible by authorized participants.

The aim of a lot of attacks is to alter code, data or messages or to replay/replicate messages or MAs. Detecting such alterations, manipulations, replays and misordering can assure the integrity of objects. Being able to establish and enforce resource constraints can prevent another big group of attacks: resource abuse and denial–of–service. The security requirement non–repudiation means that it is possible to prove that a certain subject has done a critical or sensitive action. Even a third party can prove who caused this action.

To prevent the circumvention of legal interfaces and to restrict rights the sandboxing concept is used. A sandbox is a very restricted environment for code execution which only can be left in a controlled manner.

Some attacks (e.g. manipulating an MA by an AS) seem very hard or even impossible to be prevented. If it is not possible to restrain these attacks technically an organizational solution is necessary, e.g. a trust relation between two entities that a particular kind of attack will not happen.

The following list summarizes security requirements (in bold) and attacks which can be prevented by services implementing these requirements. Some attacks are listed several times. This means either that more than one requirement covers the attack or that more services implementing the requirements are necessary to prevent a single attack.

Authentication: Masquerade, theft of rights, repudiation, replication, replay, redirection, denial–of–execution, denial–of–service, resource misuse

Authorization and Access: Theft of rights, denial–of–service, resource misuse

Confidentiality: Eavesdropping, theft of rights

Integrity: Theft of rights, replication, delay, replay, redirection, alteration, execution trace manipulation

Non–Repudiation: Repudiation

Resource–Constraints: Denial–of–service, resource misuse

Sandboxing: Circumvention Attack

5. Security Architecture

This section gives an outline of a security architecture with its major components and services. Afterwards, a description of the life cycle of a migrating agent follows to demonstrate the work of the architecture.

5.1. Components of a security architecture

The previous sections already gave an idea of necessary components and services to build a security architecture for mobile agent based management systems. Figure 4 shows this architecture with major dependencies between components during the life cycle of an agent.

It is obvious that many parts will depend on cryptographic functions based on symmetric and asymmetric keys to encrypt and sign data. Therefore, a security architecture should integrate a cryptographic library in its base services. Certainly, existing libraries should be used whenever possible. As usual, with asymmetric key functions it is necessary to have a safe and reliable key management and distribution. In order to be able to assign rights to identities there is also a need for a trust center that certifies accreditation of identities including agent systems.

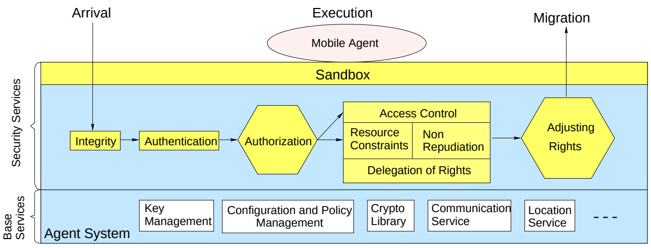

Figure 4. Security Architecture

Security must also have implications on the general design and architecture of agent systems. The agent system must be able to protect itself and access to the host system. The development of Java [9] shows that this must start with the programming language and includes concepts like sandboxes and virtual machines. They provide a locked, secured run-time environment for mobile agents that prevents any action out of control of the agent system. Each mobile agent running has its own sandbox that no other agent can directly access. Therefore, the agent system must be designed with security in mind. If not, it will not be possible to implement secure access control and security boundaries.

In order to prove integrity it is not sufficient to simply check some signatures. As mobile agents can visit several places integrity must include all computations since initiation. As any place inbetween may tamper with signatures added before, checking integrity may require doublechecking with a logging or integrity service.

Rights are the concept to properly use and exploit access control mechanisms. As already mentioned before the main components for establishing and handling rights are authentication and authorization. They depend on the basic concepts of rights like subject or object-based access rights, i.e. whether subjects carry access rights or objects have access lists. Moreover, the access control and resource constraining capabilities of the agent system must be instrumented to suit the concept of rights.

If agents are able to delegate rights everything gets more complex. With rights being 'movable', an entity may try to steal rights. This makes clear that delegation is not just a local matter between two agents and an agent system. On the contrary, an independent and trusted delegation service must probably mediate and enforce delegated rights. In addition, the conceptual design of rights must reflect delegation.

The problems of non-repudiation are similar to but more general as delegation. While delegation is only about rights non-repudiation is about any operation. These parallels may lead to some synergy between both problems. Similar to delegation non-repudiation requires a third-party non-repudiation service that records any operation that might be subject to repudiation. Moreover, it mediates in cases of repudiation proving who was responsible for actions recorded.

Configuration management of all components and services mentioned as well as of the agent system in general is an important task to establish and maintain a high level of security. Policies define a general outline of these requirements. Therefore, a policy component must enforce policies on agent systems. It parses policies and configures the system and the rights of agents.

5.2. Security Related Aspects of an Agent Life Cycle

This section outlines the life cycle of a migrating agent to give an example how components and services work together. Figure 4 depicts part of the life cycle. To begin with, when a manager initiates an agent it signs it to identify the initiator, the first agent system etc. It supplies the agent with necessary rights. If an agent system receives an agent from the communication network, some decryption and host authentication may already happen depending on the communication service. This information can influence later checks. If the agent has been encrypted the agent system decrypts it and tests the integrity of the data received by checking the signature that the sender has appended. Further checks are necessary to detect replays, redirections or replications.

After the agent system concluded these checks it can assign first security attributes to the agent received, e.g. the overall security 'level' of the transmission depending on the kind of encryption used. If possible, the agent system may already do some first code verification to assure that it is a mobile agent confirming to the implementation language specification. The following step is very important: authentication. The agent system verifies signatures and certificates attached and may find out, e.g. who wrote the agent, who sent it at the beginning, or intermediate locations. The agent system tries to map the set of identities attached to existing, valid subjects defining responsibilities for any later action. This may involve the key server or trust center.

Once authenticated, the agent system authorizes the agent, i.e. it assigns rights and checks credentials carried with the agent. As several parts influence authentication, authorization is not as easy as in ordinary operating systems. Trustworthiness may depend on each identity authenticated. Some of them may 'add' rights, e.g. agents signed by two or more managers by aggregating their rights. Others may 'remove' rights, e.g. for agents that visited a possibly malicious agent system. The actual way of determining rights may depend on security policies defined.

If the agent system decides to execute the agent after these checks, it starts the agent. In order to protect the agent system, the host system and other agents, the agent runs in a sandbox. Checks at run-time must make sure that the agent is not able to make illegal operations or violate access rights. To implement non-repudiation some operations require to log some state information to a non-repudiation service.

When the agent has finished its work and wants to migrate to another place the agent system stops execution and packs the agent with its current state. It may adjust rights of the agent, e.g. if the agent needs more rights at its next locations. The agent system signs the result to certify the execution and may log this to a non-repudiation service. Finally, it opens a (secure) communication link with the new place and sends the agent, maybe after encrypting it.

6. Conclusions

The proposed architecture is a first step towards a comprehensive, integrating security architecture for mobile agents. It needs refinement in some areas. It is necessary to integrate further experience from research and implementations to improve it.

We use this architecture as foundation for an implementation in our agent system **MASA** (Mobile Agent System Architecture). Several existing security technologies are considered for the suitability in agent systems, e.g. for confidentiality and integrity. We commenced to implement first parts like authentication. Evaluation of different concepts to model rights will follow soon. Afterwards, components depending on rights can be implemented. An authorization component must go along with access and resource control mechanisms. After completing these basic components for rights, delegation and non-repudiaton need closer attention.

Development of configuration and policy management of this security architecture should accompany all this although it needs some more considerations about management of mobile agent systems in general before.

Acknowledgment The authors would like to thank the members of the Munich Network Management (MNM) Team for helpful discussions and valuable comments on earlier versions of this paper. The MNM Team directed by Prof. Dr. Heinz-Gerd Hegering is a group of researchers at the University of Munich, the Munich University of Technology, and the Leibniz Supercomputing Center of the Bavarian Academy of Sciences. Its webserver is located at http://wwwmnmteam.informatik.uni-muenchen.de.

References

[1] S. Berkovits, J. D. Guttman, and V. Swarup. Authentication for Mobile Agents. In Vigna [27], pages 114–136.
[2] A. Bieszczad, B. Pagurek, and T. White. Mobile agents for network management. *IEEE Communication Surveys*, 1(1), 1998.
[3] D. M. Chess. Security Issues in Mobile Code Systems. In Vigna [27], pages 1–14.
[4] G. Edjlali, A. Acharya, and V. Chaudhary. History-based access control for mobile code. In *Proc. of the 5th ACM Conf. on Computer and Communications Security*. ACM, 1998.
[5] M. El-Darieby and A. Bieszczad. Intelligent Mobile Agents: Torwards Network Fault Management Automation. In Sloman et al. [24], pages 611–622.
[6] J. Feigenbaum and P. Lee. Trust Management and Proof-Carrying Code in Secure Mobile-Code Applications. In *DARPA Workshop on Foundations for Secure Mobile Code Workshop*. DARPA, 1997.
[7] M. Feridun, W. Kasteleijn, and J. Krause. Distributed Management with Mobile Components. In Sloman et al. [24], pages 857–870.
[8] G. Goldszmit and Y. Yemini. Distributed Managment by Delegation. In *Proceedings of the 15th International Conference on Distributed Computing Systems*, June 1995.
[9] L. Gong, M. Mueller, H. Prafullchandra, and R. Schemers. Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java (TM) Development Kit 1.2. In *USENIX Symp. on Internet Technologies and Systems*, 1997.
[10] M. S. Greenberg, J. C. Byington, T. Holding, and D. G. Harper. Mobile Agents and Security. *IEEE Communications Magazine*, 36(7):76–85, July 1998.
[11] H.-G. Hegering, S. Abeck, and B. Neumair. *Integrated Management of Networked Systems – Concepts, Architectures and their Operational Application*. Morgan Kaufmann Publishers, ISBN 1-55860-571-1, 1999. 651 p.
[12] F. Hohl. An Approach to Solve the Problem of Malicious Hosts. Technical Report 1997/03, Fakultät für Informatik, Universität Stuttgart, Mar. 1997.
[13] F. Hohl. Time Limited Blackbox Security: Protecting Mobile Agents from Malicious Hosts. In Vigna [27], pages 92–113.
[14] Information Technology – Open Systems Interconnection – Security Frameworks in Open Systems – Overview. IS 10181-1, International Organization for Standardization and International Electrotechnical Committee, Nov. 1995.
[15] Information Processing Systems – Open Systems Interconnection – Basic Reference Model. IS 7498, International Organization for Standardization and International Electrotechnical Committee, 1984.
[16] Information Processing Systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture. IS 7498-2, International Organization for Standardization and International Electrotechnical Committee, 1989.
[17] G. Karjoth, N. Asokan, and C. Gülcü. Protecting the Computation Results of Free-Roaming Agents. In Rothermel and Hohl [19], pages 195–207.
[18] A. Küpper and A. S. Park. Stationary vs. Mobile User Agents in Future Mobile Telecommunication Networks. In Rothermel and Hohl [19], pages 112–123.
[19] K. Rothermel and F. Hohl, editors. *Mobile Agents (MA '98)*, volume 1477 of *LNCS*, Berlin; Heidelberg, 1998. Springer.
[20] A. Sahai and C. Morin. Enabling a Mobile Network Manager (MNM) Through Mobile Agents. In Rothermel and Hohl [19], pages 249–260.
[21] T. Sander and C. F. Tschudin. Protecting Mobile Agents Against Malicious Hosts. In Vigna [27], pages 44–60.
[22] J. Schönwälder. Network Management by Delegation - From Research Prototypes Towards Standards. In *8th Joint European Networking Conf. (JENC8)*, Edinburgh, May 1997.
[23] J. Schönwälder and J. Quittek. Secure Management by Delegation within the Internet Management Framework. In Sloman et al. [24], pages 687–700.
[24] M. Sloman, S. Mazumdar, and E. Lupo, editors. *Integrated Network Management VI (IM'99)*, Boston, MA, May 1999. IEEE Publishing.
[25] W. Stallings. *Cryptography and Network Security — Principles and Practice*. Prentice Hall, 1998.
[26] G. Vigna. Cryptographic Traces for Mobile Agents. In Vign 98a [27], pages 137–153.
[27] G. Vigna, editor. *Mobile Agents and Security*, number 1419 in LNCS, Berlin, Heidelberg, 1998. Springer.